HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P. O. Box 272400
Fort Collins, Colorado 80527-2400

PATENT APPLICATION

ATTORNEY DOCKET NO. 10015199-1

IN THE
UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor(s): Peter Van Giel, et al. **Confirmation No.:** 4094

Application No.: 09/851,963 **Examiner:** Bell, Meltin

Filing Date: 05/10/2001 **Group Art Unit:** 2121

Title: A METHOD AND SYSTEM FOR AUDITING AN ENTERPRISE CONFIGURATION

Mail Stop Appeal Brief-Patents
Commissioner For Patents
PO Box 1450
Alexandria, VA 22313-1450

TRANSMITTAL OF APPEAL BRIEF

Sir:

Transmitted herewith is the Appeal Brief in this application with respect to the Notice of Appeal filed on **10/04/2004**.

The fee for filing this Appeal Brief is (37 CFR 1.17(c)) $340.00.

(complete (a) or (b) as applicable)

The proceedings herein are for a patent application and the provisions of 37 CFR 1.136(a) apply.

(X) (a) Applicant petitions for an extension of time under 37 CFR 1.136 (fees: 37 CFR 1.17(a)-(d) for the total number of months checked below:

(X)	one month	$110.00
( )	two months	$430.00
( )	three months	$980.00
( )	four months	$1530.00

12/10/2004 SSESHE1 00000028 082025 09851963
02 FC:1251 120.00 DA

( ) The extension fee has already been filled in this application.

( ) (b) Applicant believes that no extension of time is required. However, this conditional petition is being made to provide for the possibility that applicant has inadvertently overlooked the need for a petition and fee for extension of time.

Please charge to Deposit Account **08-2025** the sum of **$450.00**. At any time during the pendency of this application, please charge any fees required or credit any over payment to Deposit Account 08-2025 pursuant to 37 CFR 1.25. Additionally please charge any fees to Deposit Account 08-2025 under 37 CFR 1.16 through 1.21 inclusive, and any other sections in Title 37 of the Code of Federal Regulations that may regulate fees. A duplicate copy of this sheet is enclosed.

(X) I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, Alexandria, VA 22313-1450. Date of Deposit: **12/08/2004**

OR

( ) I hereby certify that this paper is being transmitted to the Patent and Trademark Office facsimile number _____ on _____

Number of pages:

Typed Name: **James A. Sprowl**

Signature: _____

Respectfully submitted,

Peter Van Giel, et al.

By _____

James A. Sprowl

Attorney/Agent for Applicant(s)
Reg. No. **25,061**

Date: **12/08/2004**

Telephone No.: **(847) 446-7399**

Atty. Dkt. No. 10015199-1

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BOARD OF PATENT APPEALS AND INTERFERENCES

Applicants: Peter Van Giel, et al.

Certificate of Mailing
This communication was mailed by first class mail, with sufficient postage attached, on or before: December 8, 2004.

Title: A METHOD AND SYSTEM FOR AUDITING AN ENTERPRISE CONFIGURATION

Appl. No.: 09/851,963 (Conf. No. 4904)

Filed: 05/10/2001

Examiner: Bell, Meltin

Art Unit: 2121

APPEAL BRIEF UNDER 37 C.F.R. §41.37

Mail Stop Appeal Brief - Patents
Commissioner for Patents
PO Box 1450
Alexandria, Virginia 22313-1450

Sir:

This Is an appeal from the decision of Examiner Meltin Bell, Group Art Unit 2121, mailed 6/3/2004, rejecting claims 24 to 42 in the present application and making the rejection FINAL.

Table of Contents

APPEAL BRIEF OF APPELLANTS

I. Real Party In Interest

The real party in interest is Hewlett-Packard Development Company, a Texas Limited Liability Partnership having its principal place of business in Houston, Texas.

II. Related Appeals and Interferences

None.

III. Status of Claims

Claims 1 – 23 have been cancelled.

Claims 24 – 42 stand finally rejected under 35 U.S.C. §103(a) as allegedly being unpatentable for obviousness over *Weber* (U.S. Patent No. 5,812,668) in view of *Desgrousilliers at al.* (U.S. Patent No. 5,715,373). This rejection was presented for the first time in a FINAL Office Action mailed on 6/3/2004. A response (with an amendment) to the FINAL Office Action was mailed on 9/3/2004, prompting an Advisory Action mailed on 9/16/2004. The Advisory Action entered the amendment but indicated that the response failed to place the application into condition for allowance.

Applicants respectfully request that the rejection of the claims 24 – 42 be overturned.

IV. Status of Amendments

One amendment after final was filed on 9/3/2004. It has been entered. A copy of the current version of claims 24 to 42 as amended appears in the Claim Appendix, Part VIII of this brief.

V. Summary of Claimed Subject Matter

Applicants propose grouping the claims into three groups for consideration on appeal, with both method and apparatus claims of similar and corresponding scope being included in each of the three groups of claims. Applicants ask the Board to consider separately the patentability of the claims in each of the three groups. To simplify the issues on appeal, applicants will focus upon one representative claim in each of these three groups of claims.

This is reasonable because the Examiner, in his rejection, has treated the corresponding method and apparatus claims in the same manner, in every case citing the same prior art passages against the corresponding elements of both the method and apparatus claims which correspond in scope. Accordingly, there is no reason to argue the patentability of the apparatus claims separately from that of the corresponding method claims.

Independent method claim 24 is proposed as representative of the broadest method and apparatus claims, which are all assigned to the first group. Dependent method claim 27 is proposed as representative of the method and apparatus claims in the second group, which are somewhat narrower. Dependent method claim 28 is proposed as representative of the method and apparatus claims in the third group, which are somewhat narrower still.

Accordingly, these three representative claims 24, 27, and 28 are presented and summarized below along with the remaining independent claims 30, 31, 37, 38, and 42. The discussions which follow will focus upon these three representative claims.

Summary of the Subject Matter of Independent Claim 24 (Representative of the Claims in Group 1)

24. A computerized method for auditing the software or hardware configurations of a plurality of computers {which are called "nodes" 302, 304 in Figure 2} **in one or more enterprises,** {300 in Figure 2} **comprising the steps of:**

> { **"[0062]** Enterprise. Collection of computers, software, and networking that comprises the computing environment of a business." (Page 10, lines 17-18)}

{ "**[0070]** **Node.** A node is a particular device in an enterprise, such as a server, workstation, printer, router, switch, or hub. A multiprocessor may be configured as a single node or as multiple nodes." (Page 11, lines 24-27)

in a fully automated manner, collecting {step 102 in Figure 1, step 700 in Figure 7} **from each of a plurality of networked computers** {which are called "nodes" 302 and 304 in Figure 2} **configuration information defining each computer's software configuration or hardware configuration or both;** {configuration information is collected from computers or "nodes" by "collectors" 104 in Figures 1 and 2 and is then stored in a "tracker database" 106 shown in Figures 1, 2, and 8}

{"**[0058]** Collectors. A collector is a command, or a series of commands, that causes programs installed at one or more nodes to gather configuration information about the node and that return reports defining the node's configuration. (Page 10, lines 6-8)}

{"**[0059]** Configuration. Any information specific to the static or the dynamic configuration of one or more nodes (or field computers, hardware, software, ...) at a given point in time." (Page 10, lines 9-11) }

{"**[0116]** Figure 7 illustrates the execution of collectors against the nodes 302, etc. in an enterprise 300. Execution is typically triggered by a timer 702. ... When a timer expires, the node list 406 within the collector database 310 is referenced, and the nodes [servers or work stations] are processed one at a time. At step 706, the list of collectors 408 is referred to, and a first collector is selected for execution. A command is sent to the node 302 being audited to execute the designated collector. ...

[0117] At 710, if there are more collectors, then program control branches back to step 706 where additional collectors are caused to be executed upon that same node 302. Then, at step 704, if there are more nodes 304, etc. to be audited, then program control returns to step 704 where the next node 304 is selected, and the analysis proceeds as described above.

[0118] Finally, after all the collectors 104 have been run at all the nodes 304, etc. to be audited, and all the information from the collectors 104 have been gathered, the collector output is transferred to the central tracker database 106 at a central site for further analysis." (page 24, line 14 to page 25, line 11)}

providing a plurality of analyzers {110 in Figure 1; "analyzers" are stored in an "analyzer database" 804 in Figures 2 and 8} **based on expert knowledge,** {see page 10, line 25 to page 11, line 2, Par. [0065] } **each analyzer comprising the executable program**

steps {"Java" analyzer code 2110, "C" (or "C++") analyzer code 2112, and "Perl" analyzer code 2114 in Figure 21} **needed to compute, from selected configuration information** {see definition of "configuration" presented above} **gathered from a single computer,** {"node" 302 or 304 in Figure 2} **a report identifying at least one issue** {the report is the "XML output" 2124 in Figure 21 which eventually is placed into an "issues database" 112 in Figures 1, 2, and 8} **relating to the computer;** {"node" 302 or 304 in Figure 2 }

> {"**[0055]** Analyzers. Analyzers are processes, defined by rules or programs, that analyze selected configuration information gathered by one or more collectors [from] ... one or more nodes, and that identify and report issues which may require attention." (Page 9, lines 19-21)}
>
> { "**[0121]** The first step in the analysis process is that of creating the analyzers 110 .... As shown in the figure [8], content experts 812 use analyzer creation utilities 802 to create the various documents that define the analyzers 110 and store them in the analyzer database 804. .... Each analyzer 110 focuses upon a particular issue or set of related issues that can arise within the nodes 302, etc., in the enterprise 300. Each analyzer is designed to generate an XML report whenever an issue is found to be present and in need of consideration by management. ...." (page 25, line 23 to page 26, line 3) }
>
> {"**[0067]** Issue. An issue is any matter that may need to be investigated by or reported to the management of an enterprise. An analyzer performs tests upon configuration data gathered from the [computers and other] nodes of an enterprise by collectors. Those tests determine if there are any issues that need to be drawn to management's attention." (Page 11, lines 11-15)}

defining at least one task definition {814 in Figure 8} **comprising a list of one or more of said computers** {"LIST OF NODES" at 814 in Figure 8} **and a list of one or more of said analyzers;** {"LIST OF ANALYZERS" at 814 in Figure 8} and

> **{"[0122]** Once the analyzers 110 are created and installed and the report templates and rules 116 are put in place, the system may then be called upon to do an assessment of the enterprise 300. An auditor 814, who may be an engineer or some other person desirous of learning about the condition of the nodes 302, etc. in the enterprise 300, requests an audit by using a task definition system 810 to create an assessment task. At 814, an assessment task A is shown. The assessment task 814 includes, in its definition, a list of the enterprises that are to be analyzed, a list of the nodes [including computers] at each enterprise which are to be subjected to analysis, and a list of the analysis that is to be performed in the form of the actual

names of the analyzers which are to be executed. In addition, the assessment task 814 includes a list of the reports that are to be generated following the analysis. ...." (page 26, lines 9-19) }

in a fully automated fashion, and guided by one or more of said task definitions – {814 in Figure 8}

harnessing {1500 in Figure 15; analyzer harness 806 in Figures 2, 8, and 21} **each of the task definition's** {814 in Figure 8} **listed analyzers** {"LIST OF ANALYZERS" at 814 in Figure 8; 1506 in Figure 15} **to configuration information** {see definition above} **gathered from each of the task definition's** {814 in Figure 8} **listed computers;** {"LIST OF NODES" at 814 in Figure 8; "nodes" 1514 in Figure 15}

{"**[0056]** Analyzer Harness. An analyzer harness is a framework or system which encapsulates, or wraps, and then executes an analyzer, providing the analyzer with collector data relating to a specific [computer or other] node or nodes of a specific enterprise each time an analyzer is encapsulated or wrapped and then executed. " (Page 9, lines 22-25)}

{"**[0065]** Framework. This is a symbolic structure in which expert knowledge and experience is placed such that it may be automatically used to analyze enterprise configuration information gathered by collectors to identify issues that may require management attention. More specifically, in the context of the preferred embodiment, expert system rules called analyzers are placed into a harness or framework and supplied with configuration information from enterprise [computers and other] nodes and are thereby enabled to report any issues that may have arisen within the enterprise." (Page 10, line 25 to Page 11, line 2)}

{"**[0123]** Once a task 814 is defined and initiated, the list of enterprises, nodes, and analyzers are passed to the analyzer harness 806. The analyzer harness 806 then proceeds by picking up the analyzers 110 from the database 804, one at a time, and with each analyzer 110 the analyzer harness 806 proceeds through the nodes 302, etc. one at a time. For each node, the harness 806 creates a framework linking the analyzer 110 to configuration information files that are retrieved from the tracker database 106. ..." (page 26, lines 22 to 28)}

processing (108 in Figure 1, and 1518 in Figure 15) **the configuration information** {"collector reports" 1516 in Figure 15 contain the configuration information collected from the computers in the enterprise} **so harnessed under the**

guidance of each analyzer's executable program steps; {2110, 2112, or 2114 in Figure 21} **and**

{"**[0123]** .... Using this framework, the harness 806 wraps the analyzer 110 in this environment and causes it to be executed in the context of the list of configuration information files that contain configuration information gathered from the [computer or other] node 302 that is being currently analyzed. ..." (page 26, lines 28- 31) }

{"**[0187]** In Figure 21, ... The analyzer loader 2102 ... calls upon an argument manager 2106 to reach out into the tracker database 106 to find collector reports 2108 and to arrange for those reports ... to be passed as incoming arguments to the analyzers 110 when they are executed by the analyzer loader 2102. The loader 2102, being written in "java", executes "java" executable analyzers 2110 directly. It calls upon a standard operating system "shell" utility to execute analyzers 2112 written in the "C" or "C++" programming languages. It calls upon an operating system interpreter to interpret and execute analyzer[s] written in "perl" or "kshell" or other interpretive languages. The output of an analyzer is shown to be an XML output at 2124 ... generated by ... sub-routines 2116 ... called by the analyzer 110 ... to generate XML "issues" output messages ...." (Page 43, line 27; Page 44, lines 4- 16)}

utilizing {114 and 118 in Figure 1} **any issue identifying report generated during the processing step for audit purposes.**

{"**[0057]** Audit. The term is normally used for a formal examination or verification of financial accounts[;] ... in this context it means a very thorough examination or verification. ..." (Page 9, line 26 to page 10, line 1)}

{"[0123] ... During its execution, the analyzer 110 calls upon special subroutines that generate short XML reports of any issue which warrants management attention and also of any error condition which may arise. After the analyzer 110 terminates, the analyzer harness 806 takes these small issue XML reports and expands them ... and creates an expanded XML report which is stored in the (XML) issues database 112 after the analysis have been run against all of the nodes 302, etc. In this manner, an extended issue report is generated in an XML format that is both human readable and also that lends itself to being incorporated into a database for automated retrieval and manipulation.

[0124] The list of reports from the task definition 814 is passed to the report generator 206. ... ... [T]he report generator 206 prepares a variety of reports, as has been explained, setting forth the status of the

enterprise 300 and its nodes 302, etc. These are then fed to various recipients of the reports 817." (Page 26, line 31 to page 27, line 20)}

Summary of the Subject Matter of Dependent Claim 27

(Representative of the Claims in Group 2)

27. A method in accordance with claim 24 wherein the processing step further includes generating, along with at least some issue identifying reports that are generated, the identity of the computers whose configuration information was processed to generate those reports {in step 222 (Figure 24), the "issue list" is "augmented" with the "node name," which is the identity of the computer whose configuration information was just analyzed} **such that both the issue identifying reports and the identity of the computers may be utilized.**

{"**[0123]** ... *After the analyzer 110[Figure 1] terminates, the analyzer harness 806[Figures 2, 8, and 21 takes these small issue XML reports and expands them, using ... information as to the identity of the[computer or other] node ... and creates an expanded XML report* which is stored in the (XML) issues database 112 [Figures 1, 2, and 8] after the analysis have been run against all of the nodes 302, etc. [Figure 2] In this manner, an extended issue report is generated in an XML format that is both human readable and also that lends itself to being incorporated into a database for automated retrieval and manipulation." (*Emphasis added* – page 27, lines 2 to 10)

{"**[0187]** The output of an analyzer is shown to be an XML output at 2124 [Figure 21] ... generated by ... sub-routines 2116 ... called by the analyzer 110 ... to generate XML "issues" output messages. *The output 2124 does not include identification of the relevant [computer or other] node.... Accordingly, the analyzer loader 2102 accepts the XML output of the analyzers 2124 and generates an expanded XML report 2126 that include[s] ... the identification of the [computer or other] node being processed....* This may be seen in Appendix F, where the XML output of an analyzer 110 [Figure 1] is shown in Part 1 [Page 65, lines 2-24 – this raw analyzer output does not include any specific computer identification], and the XML report 2126 of the analyzer loader 2102 is shown in Part 2 [Page 65, line 26 to page 68, line 3 – this augmented analyzer output does include the added computer identification or name "dineruat" on line 5, page 66 and again on line 5, page 67]." (*Emphasis added* – page 44, lines 13-25)}

"**[0195]** At step 2222 [Figure 24], the issue list or failure list XML output (See Sample 1 in Appendix F [Page 65, lines 2-24 – this raw analyzer output does not include any specific computer identification]) is captured and is returned to the analyzer loader 2102 in the form of an XML output 2124. *At step 2222, the analyzer loader 2102 expands the size of this XML output by adding to it information defining the name of the[computer or other] node 302 [Figure 2] from which the collector configuration information came,* the name of the assessment task 814 [Figure 8] launched by the auditor 813, and other such background information. ... The analyzer loader 2102 [Figure 21] then generates a new segment for a growing output XML report 2126 that includes all of the above added information added to the analyzer XML output 2124. A sample of such a report can be seen in Part 2 of Appendix F [Page 65, line 26 to page 68, line 3 – this augmented analyzer output does include the added computer identification or name "dineruat" on line 5, page 66 and again on line 5, page 67]." (*Emphasis added* – Page 46, line 9 to 24)}

Summary of the Subject Matter of Dependent Claim 28

(Representative of the Claims in Group 3)

28. A method in accordance with claim 27 which further includes:

providing a plurality of audit report templates; {116 in Figure 1 and 204 in Figure 8}

{"**[0078]** Now, the issues stored in the issues database 112 may be used in another way to review the overall performance of the enterprise. To this end, in step 114 the issues are analyzed using rules written by the experts, and *a report is generated as desired by the auditor. Generally speaking, the reports are generated from templates stored in the report templates and rules database 204. The reports may be presented in step 118 to enterprise management, technical management, the field engineering team,* and to a workflow system or healer system (self-healing technology). ..." (*Emphasis added* – page 13, lines 11 to 18)}

defining the at least one task definition {814 in Figure 8} **to further comprise a list of one or more audit report templates;** {"LIST OF REPORTS" within element 814 in Figure 8} **and**

{"**[0122]** Once the analyzers 110 are created and installed and the report templates and rules 116 are put in place, the system may then be

called upon to do an assessment of the enterprise 300. An auditor 814, who may be an engineer or some other person desirous of learning about the condition of the nodes 302, etc. in the enterprise 300, requests an audit by using a task definition system 810 to create an assessment task. At 814, an assessment task A is shown. The assessment task 814 includes, in its definition, a list of the enterprises that are to be analyzed, a list of the nodes at each enterprise which are to be subjected to analysis, and a list of the analysis that is to be performed in the form of the actual names of the analyzers which are to be executed. *In addition, the assessment task 814 includes a list of the reports that are to be generated following the analysis.* Report generation may be done at the time of the analysis, or the reports may be generated at a later time in a separate session." (*Emphasis added* – page 26, lines 9-21)}

following the storing step, and guided by the task definition's listed audit report templates {"LIST OF REPORTS" within element 814 in Figure 8} **and by any issue identifying report generated during the processing step,** {report 2126 in Figure 21, placed in the issues database 112 in Figures 1, 2, and 8} **generating** {114 in Figure 1} **one or more audit reports.** { 208 in Figure 2; 817 in Figure 8}

{"[0124] *The list of reports from the task definition 814 [Figure 8] is passed to the report generator 206. The report generator 206 also has access to the report templates and rules database 204 and to the XML issue report which can be retrieved from the (XML) issues database 112.* Using all of these materials, an expert system engine within, or supplementing, the report generator 206 evaluates the rules and, under their guidance, examines the issue information, generating high-level conclusions for management concerning the general state of the enterprise. *Then, using the report templates, the report generator 206 prepares a variety of reports, as has been explained, setting forth the status of the enterprise 300 and its nodes 302, etc.[Figure 2] These are then fed to various recipients of the reports 817 [Figure 8].*" (*Emphasis added* – page 27, lines 11 to 20)

Summary of the Subject Matter of Independent Claim 30

{Note: Claim 30 is a narrow method claim assigned to Group 3, and accordingly all of the passages from the specification quoted above with respect to claims 24, 27, and 29 are relevant to understanding claim 30. Those passages are incorporated here by reference.}

30. A computerized method for auditing the software or hardware configurations of a plurality of computers {these are called "nodes" 302, 304 in Figure 2 } **in one or more enterprises** {300 in Figure 2 }**, comprising the steps of:**

in a fully automated manner, collecting {step 102 in Figure 1, step 700 in Figure 7} **from each of a plurality of networked computers** {"nodes" 302 and 304 in Figure 2} **configuration information** {stored in a "tracker database" 106 shown in Figures 1, 2, and 8} **defining each computer's software configuration or hardware configuration or both** {stored in a "tracker database" 106 shown in Figures 1, 2, and 8} **and placing this configuration information into a tracker database;** {106 shown in Figures 1, 2, and 8}

providing a plurality of analyzers {110 in Figure 1; "analyzers" are stored in an "analyzer database" 804 in Figures 2 and 8} **based on expert knowledge,** {see page 10, line 25 to page11, line 2, Par. [0065] } **each analyzer comprising the executable program steps** {2110, 2112, and 2114 in Figure 21} **needed to compute, from selected configuration information** {see definition above} **gathered from a single computer,** {"node" 302 or 304 in Figure 2 } **a report identifying at least one issue** {"XML output" 2124 in Figure 21 which eventually is placed into an "issues database" 112 in Figures 1, 2, and 8} **relating to the computer,** {"node" 302 or 304 in Figure 2 } **and placing these analyzers into an analyzer database;** {112 in Figures 1, 2, and 8}

providing a plurality of audit report templates, {116 in Figure 1 and 204 in Figure 8; page 28} **and placing these templates into a report template database;** {204 in Figures 2 and 8}

defining at least one task definition {814 in Figure 8} **comprising a list of one or more of said computers,** {"LIST OF NODES" at 814 in Figure 8} **a list of one or more of said analyzers,** {"LIST OF ANALYZERS" at 814 in Figure 8} **and a list of one or more of said audit report templates;** {"LIST OF REPORTS" AT 814 in Figure 8} **and**

in a fully automated fashion, and guided by one or more of said task definitions {814 in Figure 8} --

harnessing {1500 in Figure 15; analyzer harness 806 in Figures 2, 8, and 21} **each of the task definition's** {814 in Figure 8} **listed analyzers** {"LIST OF ANALYZERS" at 814 in Figure 8; 1506 in Figure 15} **to configuration information** {see definition above} **gathered from each of the task definition's** {814 in Figure 8} **listed computers;** {"LIST OF NODES" at 814 in Figure 8; "nodes" 1514 in Figure 15}

processing (108 in Figure 1, and 1518 in Figure 15) **the configuration information** {"collector reports" 1516 in Figure 15 contain the configuration information collected from the computers in the enterprise} **so harnessed under the guidance of each analyzer's executable program steps;** {2110, 2112, or 2114 in Figure 21}

storing any issue identifying reports generated during the processing step in an issues database {112 in Figures 1, 2, and 12} **together with, in at least some instances, the identity of the computers whose configuration information was processed to generate these reports;** {222 in Figure 24, where the "issue list" is "augmented" with the "node name," which is the identity of the computer} **and**

guided by the task definition's listed audit report templates {"LIST OF REPORTS" within 814 in Figure 8} **and by any issue identifying reports and computer identities stored in the issues database,** {report 2126 in Figure 21, placed in the issues database 112 in Figures 1, 2, and 8} **generating** {step 114 in Figure 1} **at least one or more audit reports.** { 208 in Figure 2; 817 in Figure 8}

Summary of the Subject Matter of Independent Claim 31

{Note: Claim 31 is a broad apparatus claim assigned to Group 1, and accordingly all of the passages from the specification quoted with respect to claim 24 (above) are relevant to understanding claim 31. Those passages are incorporated here by reference.}

31. A computerized system for auditing the software or hardware configurations of a plurality of computers in one or more enterprises, the system comprising:

a plurality of computers {these are called "nodes" 302, 304 in Figure 2} **each computer having software and hardware components configured in a variety of measurable ways;**

a network {323 in Figure 2 (added by amendment); page 14, lines 16 to 17, Par. [0081] (amended to read "The analyzer server is linked to the enterprise 300 via an ISDN line or some other form of wide area network 323 ...." } **interconnecting said computers** {"nodes" 302, 304 in Figure 2};

a plurality of collectors {104 in Figures 1 and 2}**, installed on at least one computer** {"nodes" 302, 304 in Figure 2}**, gathering from said computers configuration information defining each computer's hardware configuration or software configuration or both;** {see definition of "configuration" presented above}

a plurality of analyzers, {110 in Figure 1} **each analyzer comprising the executable program steps** {2110, 2112, and 2114 in Figure 21} **needed to compute, from selected configuration information** {see definition above} **gathered from a single computer,** {"node" 302 or 304 in Figure 2 } **a report identifying at least one issue** {"XML output" 2124 in Figure 21 which eventually is placed into an "issues database" 112 in Figures 1, 2, and 8} **relating to the computer;** {"node" 302 or 304 in Figure 2 }

at least one machine-readable task definition {814 in Figure 8; page 23} **comprising a list of one or more computers** {"LIST OF NODES" at 814 in Figure 8} **and a list of one or more analyzers;** {"LIST OF ANALYZERS" at 814 in Figure 8} **and**

an analyzer harness {806 in Figures 2, 8, and 21} **guided by said task definition** {814 in Figure 8} **through the process of sequentially harnessing** {1500 in Figure 15} **each of the task definition's** {814 in Figure 8} **listed analyzers** {"LIST OF

ANALYZERS" at 814 in Figure 8; 1506 in Figure 15} **to configuration information gathered from each of the task definition's** {814 in Figure 8} **listed computers** ("LIST OF NODES" at 814 in Figure 8; "nodes" 1514 in Figure 15) **and executing** (108 in Figure 1, and 1518 in Figure 15) **each analyzer's executable program steps** {2110, 2112, or 2114 in Figure 21} **upon the harnessed configuration information,** {"collector reports" 1516 in Figure 15 contain the configuration information collected from the computers in the enterprise} **thereby generating at least some issue identifying reports.** {114 and 118 in Figure 1}

Summary of the Subject Matter of Independent Claim 37

{Note: Claim 37 is a narrow apparatus claim assigned to Group 3, and accordingly all of the passages from the specification quoted above with respect to claims 24, 27, and 29 are relevant to understanding claim 37. Those passages are incorporated here by reference.}

37. A computerized system for auditing the software or hardware configurations of a plurality of computers in one or more enterprises, the system comprising:

a plurality of computers {these are called "nodes" 302, 304 in Figure 2} **each computer having software and hardware components configured in a variety of measurable ways;**

a network {323 in Figure 2 (added by amendment); page 14, lines 16 to 17, Par. [0081] (amended to read "The analyzer server is linked to the enterprise 300 via an ISDN line or some other form of wide area network 323 ...." } **interconnecting said computers** {"nodes" 302, 304 in Figure 2} **with a tracker database** {106 in Figures 1, 2, and 8}**;**

a plurality of collectors, {104 in Figures 1 and 2} **installed on at least one computer,** {"nodes" 302, 304 in Figure 2} **gathering from said computers configuration information defining each computer's hardware configuration or software**

configuration or both, {see definition of "configuration" presented above} **and storing the gathered configuration information in the tracker database;** {106 in Figures 1, 2, and 8}

a plurality of analyzers {110 in Figure 1; "analyzers" are stored in an "analyzer database" 804 in Figures 2 and 8} **residing within an analyzer database,** {804 in Figures 2 and 8} **each analyzer comprising the executable program steps** {2110, 2112, and 2114 in Figure 21} **needed to compute, from selected configuration information** {see definition above} **gathered from a single computer,** {"nodes" 302, 304 in Figure 2} **a report identifying at least one issue** {"XML output" 2124 in Figure 21 which eventually is placed into an "issues database" 112 in Figures 1, 2, and 8} **relating to the computer;** {"nodes" 302, 304 in Figure 2}

audit report templates {116 in Figure 1} **residing within a report templates database;** {204 in Figure 2 and 8}

at least one machine-readable task definition {814 in Figure 8} **comprising a list of one or more computers,** {"LIST OF NODES" at 814 in Figure 8} **a list of one or more analyzers,** {"LIST OF ANALYZERS" at 814 in Figure 8} **and a list of one or more audit report templates;** {"LIST OF REPORTS" AT 814 in Figure 8}

an analyzer harness {806 in Figures 2, 8, and 21} **connecting to said tracker database** {106 in Figures 1, 2, and 8} **and to said analyzer database** {804 in Figures 2 and 8} **and guided by said task definition** {814 in Figure 8} **through the process of sequentially harnessing** {1500 in Figure 15} **each of the task definition's** {814 in Figure 8} **listed analyzers** {"LIST OF ANALYZERS" at 814; 1506 in Figure 15} **to configuration information** {see definition above} **gathered from each of the task definition's** {814 in Figure 8} **listed computers,** {"LIST OF NODES" at 814; "nodes" 1514 in Figure 15} **executing** {108 in Figure 1, and 1518 in Figure 15} **each analyzer's executable program steps** {2110, 2112, or 2114 in Figure 21} **upon the harnessed configuration information,** {"collector reports" 1516 in Figure 15 contain the configuration information collected from the computers in the enterprise} **and storing any issue identifying reports** {114 and 118 in Figure 1} **generated during this execution in an issues database** {112 in Figures 2 and 8} **together with, in at least some instances, the identity of the computers** {222 in Figure 24,

where the "issue list" is "augmented" with the "node name," which is the identity of the computer} **whose configuration information was harnessed to generate these issue identifying reports; and**

a report generator {206 in Figures 2 and 8} **connecting to said issues database** {112 in Figures 1, 2, and 8} **and to said report template database** {204 in Figures 2 and 8} **and guided by said task definition** {814 in Figure 8} **through the process of transforming at least one of the task definition's listed audit report templates** {"LIST OF REPORTS" which is part of task definition 814 in Figure 8} **into one or more audit reports, guided by any issue identifying reports** {in the issues database 112 in Figures 1, 2, and 8} **and computer identities** {222 in Figure 24, where the "issue list" is "augmented" with the "node name," which is the identity of the computer} **stored in the issues database.** {112 in Figures 1, 2, and 8}

Summary of the Subject Matter of Independent Claim 38

{Note: Claim 38 is a broad "means for" apparatus claim assigned to Group 1, and accordingly all of the passages from the specification quoted above with respect to claim 24 are relevant to understanding claim 38. Those passages are incorporated here by reference.}

38. A computerized system for auditing the software or hardware configurations of a plurality of computers in one or more enterprises, the system comprising:

a plurality of computers {these are called "nodes" 302, 304 in Figure 2} **each computer having software and hardware components configured in a variety of measurable ways;**

tracker database means {tracker database 106 in Figures 1, 2, and 8}, **analyzer database means** {analyzer database 804 in Figures 2 and 8}, **and issues database means** {issues database 112 in Figures 1, 2, and 8} **all for storing digital information;**

a network {323 in Figure 2 (added by amendment); page 14, lines 16 to 17, Par. [0081] (amended to read "The analyzer server is linked to the enterprise 300 via an ISDN line or some other form of wide area network 323 ...." } **interconnecting said computers** {"nodes" 302, 304 in Figure 2} **with said tracker database means;** {tracker database 106 in Figures 1, 2, and 8}

collector means, {collectors 104 in Figures 1 and 2} **installed on at least one computer,** {"nodes" 302, 304 in Figure 2} **for gathering from said computers** {"nodes" 302, 304 in Figure 2} **configuration information defining each computer's hardware configuration or software configuration information or both,** {see definition above} **and for storing this gathered configuration information in said tracker database means;** {tracker database 106 in Figures 1, 2, and 8}

analyzer means, {110 in Figure 1; "analyzers" are stored in an "analyzer database" 804 in Figures 2 and 8} **residing within said analyzer database means,** {analyzer database 804 in Figures 2 and 8} **for defining the executable program steps** {2110, 2112, and 2114 in Figure 21} **needed to compute, from selected configuration information** {see definition above} **gathered from a single computer,** {"node" 302 or 304 in Figure 2} **a report identifying at least one issue** {"XML output" 2124 in Figure 21 which eventually is placed into an "issues database" 112 in Figures 1, 2, and 8} **relating to the computer;** {"node" 302 or 304 in Figure 2 }

machine-readable task definition means {814 in Figure 8} **for defining a list of one or more computers** {"LIST OF NODES" at 814 in Figure 8} **and a list of one or more analyzer means;** {"LIST OF ANALYZERS" at 814 in Figure 8} **and**

analyzer harness means {806 in Figures 2, 8, and 21} **connecting to said tracker database means,** {tracker database 106 in Figures 1, 2, and 8} **said analyzer database means,** {analyzer database 804 in Figures 2 and 8} **and said issues database means** {issues database 112 in Figures 1, 2, and 8} **for sequentially harnessing,** {1500 in

Figure 15} **under the guidance of said task definition means,** {task definition 814 in Figure 8} **each of the task definition mean's listed analyzer means** {"LIST OF ANALYZERS" at 814 in Figure 8; 1506 in Figure 15} **to configuration information** {see definition above} **gathered from each of the task definition mean's listed computers,** {"LIST OF NODES" at 814 in Figure 8; "nodes" 1514 in Figure 15} **for executing** {108 in Figure 1, and 1518 in Figure 15} **each analyzer mean's executable program steps** {2110, 2112, or 2114 in Figure 21} **upon the harnessed configuration information,** {"collector reports" 1516 in Figure 15 contain the configuration information collected from the computers in the enterprise} **and for storing any issue identifying report** {114 and 118 in Figure 1} **generated during this execution in said issues database** {issues database 112 in Figures 1, 2, and 8}.

Summary of the Subject Matter of Independent Claim 42

{Note: Claim 42 is a narrow "means for" apparatus claim assigned to Group 3, and accordingly all of the passages from the specification quoted above with respect to claims 24, 27, and 29 are relevant to understanding claim 42. Those passages are incorporated here by reference.}

42. A computerized system for auditing the software or hardware configurations of a plurality of computers in one or more enterprises, the system comprising:

a plurality of computers {these are called "nodes" 302, 304 in Figure 2} **each computer having software and hardware components configured in a variety of measurable ways;**

tracker database means, {tracker database 106 in Figures 1, 2, and 8} **analyzer database means,** {analyzer database 804 in Figures 2 and 8} **report template database means** {report templates and rules database 204 in Figures 2 and 8}, **and issues**

database means {issues database 112 in Figures 1, 2, and 8} **all for storing digital information;**

a network {323 in Figure 2 (added by amendment); page 14, lines 16 to 17, Par. [0081] (amended to read "The analyzer server is linked to the enterprise 300 via an ISDN line or some other form of wide area network 323 ...." } **interconnecting said computers** {"nodes" 302, 304 in Figure 2} **with said tracker database means;** {tracker database 106 in Figures 1, 2, and 8}

collector means, {collectors 104 in Figures 1 and 2} **installed on at least one computer,** {"nodes" 302, 304 in Figure 2} **for gathering from said computers** {"nodes" 302, 304 in Figure 2} **configuration information defining each computer's hardware configuration or software configuration or both,** {see definition presented above} **and for storing this gathered configuration information in said tracker database means;** {tracker database 106 in Figures 1, 2, and 8}

analyzer means, {110 in Figure 1; "analyzers" are stored in an "analyzer database" 804 in Figures 2 and 8} **residing within said analyzer database means,** {analyzer database 804 in Figures 2 and 8} **for defining the executable program steps** {2110, 2112, and 2114 in Figure 21} **needed to compute, from selected configuration information** {see definition above} **gathered from a single computer,** {"node" 302 or 304 in Figure 2} **a report identifying at least one issue** {"XML output" 2124 in Figure 21 which eventually is placed into an "issues database" 112 in Figures 1, 2, and 8} **relating to the computer;** {"node" 302 or 304 in Figure 2}

audit report template means, {report templates and rules 116 in Figure 1} **residing within said report template database means,** {report templates and rules database 204 in Figure 2 and 8} **for defining all or portions of one or more audit reports** { 208 in Figure 2; 817 in Figure 8};

machine-readable task definition means {814 in Figure 8} **for defining a list of one or more computers,** {"LIST OF NODES" at 814 in Figure 8} **a list of one or more analyzer means,** {"LIST OF ANALYZERS" at 814 in Figure 8} **and a list of one or more audit report template means;** {"LIST OF REPORTS" at 814 in Figure 8}

analyzer harness means {806 in Figures 2, 8, and 21} **connecting to said tracker database means,** {tracker database 106 in Figures 1, 2, and 8} **said analyzer database means,** {analyzer database 804 in Figures 2 and 8} **and said issues database means** {issues database 112 in Figures 1, 2, and 8} **for sequentially harnessing** {1500 in Figure 15}, **under the guidance of said task definition means,** {task definition 814 in Figure 8} **each of the task definition mean's listed analyzer means** {"LIST OF ANALYZERS" at 814 in Figure 8; 1506 in Figure 15} **to configuration information** {see definition above} **gathered from each of the task definition mean's listed computers,** {"LIST OF NODES" at 814 in Figure 8; "nodes" 1514 in Figure 15} **for executing** {108 in Figure 1, and 1518 in Figure 15} **each analyzer mean's executable program steps** {2110, 2112, or 2114 in Figure 21} **upon the harnessed configuration information,** {"collector reports" 1516 in Figure 15 contain the configuration information collected from the computers in the enterprise} **and for storing any issue identifying reports** {114 and 118 in Figure 1} **generated during this execution to in said issues database** {issues database 112 in Figures 1, 2, and 8} **together with, in at least some instances, the identity of the computers** {see step 222 in Figure 24, where the "issue list" is "augmented" with the "node name," which is the identity of the computer} **whose configuration information** {"collector reports" 1516 in Figure 15 contain the configuration information collected from the computers in the enterprise} **was harnessed to generate these issue identifying reports;** {114 and 118 in Figure 1} **and**

report generator means, {report generator 206 in Figures 2 and 8} **connecting to said issues database means** {issues database 112 in Figures 1, 2, and 8} **and said report template database means,** {report template and rules database 204 in Figures 2 and 8} **for generating at least one audit report, guided by the task definition mean's listed report templates** {"LIST OF REPORTS" which is part of task definition 814 in Figure 8} **and by the issue identifying reports** {in the issues database 112 in Figures 1, 2, and 8} **and computer identities** {222 in Figure 24, where the "issue list" is "augmented" with the "node name," which is the identity of the computer} **stored in the issues database.** {112 in Figures 1, 2, and 8}

VI. Grounds for Rejection

Claims 24 to 42 stand rejected under 35 U.S.C. §103(a) as allegedly being unpatentable for obviousness over *Weber* (U.S. Patent No. 5,812,668) in view of *Desgrousilliers at al.* (U.S. Patent No. 5,715,373).

VII. Argument – Claims 24 to 42 Are Patentable

A. Grouping of the Claims Into Three Groups of Related Claims for Consideration on Appeal

The grouping together of claims having similar scopes proposed for consideration on this appeal is as follows:

Group 1 – Claims 24-25, 31-32, 38, and 40 grouped together. Claim 24 is suggested as a claim representative of this group of claims.

Group 2 – Claims 26-27, 33-34, and 39 are grouped together. Claim 27 is suggested as a claim representative of this group of claims.

Group 3 – Claims 28-30, 35-37, and 41-42 are grouped together. Claim 28 is suggested as a claim representative of this group of claims.

This application includes method claims 24 - 30, apparatus claims 31 - 37, and "means for" apparatus claims 38 – 42. Each of these sets of claims includes one broad, independent claim (the first claim in each set); one narrow, independent claim (the last claim in each set); and a range of dependent claims of varying scope.

To simplify the issues on appeal and to shorten this brief, applicants elect and propose to group together the claims of corresponding scope into three groups each containing claims taken from each of these three sets of claims. Each group of claims accordingly includes two method claims, two apparatus claims of similar scope to the method claims, and one or two "means for" apparatus claims, also of similar scope to the method claims.

Applicant suggests to the Board of Patent Appeals that the method claims 24, 27, and 28 are representative of the remaining claims in each of these three groups. Accordingly, the

detailed claim discussions presented in section H below will focus upon the specific wording of the representative method claims 24, 27, and 28.

B. Organization of the Arguments Presented Below

Section C presents a concise description of the present invention, as defined by the claims in the three groups.

Sections D - G below present general reasons, applicable to all of the claims, as to why the claims on appeal are allowable over the combination of the *Weber* and *Desgrousilliers, et al.* patents. Section H then discusses the individual claims, element by element, as well as the prior art passages and drawings which the Examiner has cited.

Sections D and E demonstrate that neither of these two patents is relevant to the present invention. Section D explains the *Weber* patent and section E explains the *Desgrousilliers, et al.* patent.

Section F demonstrates that there is no motivation to combine the teachings of these two patents, since they relate to entirely different fields of technology, and thus a legal case of obviousness under 35 U.S.C. Section 103 cannot be made out.

Section G explains that the teachings of these two patents require the removal from normal service of a computer system that is to be tested, while the present invention is directed to testing carried out while the computer systems are operating normally.

Section H presents a detailed, element-by-element discussion of the representative claims taken from each of the three groups of claims. The elements (other than the claim preambles) of the representative claims 24, 27, and 28 are discussed individually and are distinguished from the specific prior art patent passages and figures which the Examiner has cited against specific claim elements. This discussion will demonstrate that none of the claim elements is taught by the two cited prior-art patents.

C. Concise Description of the Present Invention

For purposes of this appeal, the invention is best illustrated in Figure 1 (overview of the method), Figure 2 (overview of the enterprise computers or "nodes" which are linked to a

central analyzer and report generator), Figure 8 (the analyzer and report generator), and Figure 21 (the analyzer harness 806 and the analyzer executable code 2110, 2112, and 2114). The claims in group 2 also require brief discussion of the step 222 shown in Figure 24 (specifically, the mention in that step of the addition of a computer's identity, or "node name," to the critical issues list generated by the analysis process so that reports may identify the names of the computers which gave rise to each critical issue).

With reference to Figure 2, the present invention, as defined in the claims of Group 1, claims 24-25, 31-32, 38, and 40, teaches collecting information on how all the computers ("nodes 302 and 304") in an enterprise 300 are configured. This computer configuration information is gathered into a "tracker database 106." The configuration information is next analyzed by analyzers that include executable code under the guidance of an analyzer harness 806. If critical "issues" that need to be reported to management are detected, reports of these issues are placed into an issues database 112. Then a report generator 206 generates various reports 208 for management, guided by templates. The steps of this process are illustrated in overview in Figure 1.

The claims in Group 1 require the analysis to be carried out by "analyzers" (a term defined in paragraph 55 – page 9, lines 19-21) that are written by, for example, computer maintenance field engineers specifically to test configuration information representing the state of selected software or hardware components of a single computer.

When enterprise computers are to be audited, Figure 8 illustrates that an auditor 813 prepares an assessment task 814 which specifies the computers (or "nodes") whose configuration information is to be analyzed. The assessment task also specifies which analyzers are to be used and which reports are to be generated following analysis.

An analyzer harness 806, the details of which are shown in Figure 21, then harnesses each specified analyzer (for example, the executable programs 2110, 2112, and 2114 shown in Figure 21) to the configuration information reports 2108 collected from each specified computer, thereby enabling the specified analyzers to detect and to report any critical issues arising on any of the specified computers – issues that may warrant or require management's attention. In this manner, hundreds or even thousands of individual computers may be subjected to individual analysis by dozens of individual, very simple analyzer programs

written in various programming languages by various field engineers. Reports 2126 of issues arising are passed on to the report generator 206 (Figures 2 and 8). Accordingly, all unusual or reportable events of interest to an auditor (for example, a maintenance engineer) are then reported in well organized reports.

The claims in Group 2, Claims 26-27, 33-34, and 39, all include an additional significant limitation that requires the "identity of the computers whose configuration information was processed" to be included with at least some of the "issue identification reports" generated by the analyzers. As is shown in the step 2222 in Figure 24, the analyzer harness 806 (and, more specifically, the analyzer loader 2102 shown in Figure 21) takes the issue list generated by analyzers (output 2124 in Figure 21) and "augments" this issue list with the "node name," which is the name of the computer giving rise to the critical issue. Accordingly, the report generator is able to identify the computers giving rise to critical issues even though the very simple analyzer programs (written, for example, in java 2110, C 2112, or Perl 2114 in Figure 21) may not even be aware of the computer names and therefore cannot identify individual computers in their issue reports. The claims in Group 2 require the analyzer harness to work in this manner, supplementing the simple analyzer issue reports with computer names, and thereby keeping the analyzers quite simple and easy to write.

The claims in Group 3, claims 28-30, 35-37, and 41-42 are more specific as to the details of the report generating process. They specifically call for the issue identification reports to be fed into a report generator 206 (Figures 2 and 8) that is also fed templates for the reports specified in the assessment task 814. These templates 116 (Figure 1) are stored in a report templates database 204 (Figures 2 and 8). The report templates can be created by the same field engineers who write the analyzers, or they may be created by others who are not necessarily programmers. Different report templates can generate widely differing reports suited to different audiences – e.g.: enterprise management, technical management, and the field engineering team, as is indicated at 118 in Figure 1.

All the claims in Group 3 also include all the limitations of the claims in Group 2, namely, the requirement that the names of the individual computers (or "nodes") giving rise to critical issue reports be added to the reports of issues generated by the analyzers.

D. The *Desgrousilliers, et al.* Patent Is Not Relevant To The Present Invention. *Desgrousilliers, et al.* Relates To A Very Different Area of Technology – Testing a Program's Requirements Prior to Writing the Actual Program

The *Desgrousilliers, et al.* patent's teachings are unrelated to the teachings of the present invention. The *Desgrousilliers, et al.* patent relates to the testing of a program's design requirements prior to writing (or coding) the actual program, not to analyzing hardware and software configuration information collected from fully programmed computers installed and operating normally in their normal operating environments.

The *Desgrousilliers, et al.* patent discloses method steps and system elements that may be used to test a computer program's design prior to the actual writing of the computer program. The patent teaches a programmer how to develop and then test a set of detailed requirements for a computer program long before the program is actually coded. This testing of requirements (not the testing of an actual program in operation) is carried out in a sheltered software development environment, and not in the field. Contrary to all this, the present application discloses method steps and system elements (recited in claims 24 to 42) that may be used to collect and then analyze data gathered from fully programmed computers operating normally in the field.

More specifically, the *Desgrousilliers, et al.* patent teaches how the process of designing a type of computer program that the patent calls a "network management application" and also calls a "subsystem control facility (SCF)" program (abstract, lines 2-4) can be improved by developing and then testing a set of requirements prior to writing (or coding) the actual program. With reference to Figure 8 of the patent, after a programmer has completed the initial design of an SCF program by developing the set of requirements, and before the SCF program is actually coded, the programmer prepares a "set of requirements for the proposed SCF" program that are "embodied in the form of a binary table [21 in Figure 8] which contains the subsystem control facility [SCF] command and display specifications" (col. 10, lines 52-58). These requirements are then "loaded into a test SCF unit 25" (col. 11, line 33) which tests the requirements under the guidance of test scripts (22 in Figure 8 and 14 in Figure 1) A "... knowledgebase development system 10 [shown in Figure 1 is] used [by the programmer] to generate test scripts [14, 28] ...." (col. 5, lines 27-31) After the SCF unit completes the testing of the requirements, the "test results files [are placed] in storage device

29" and may then be "manually analyzed" by the programmer (col. 13, lines 55-58), or they may "be analyzed by inference engine 16 [Figure 1] ...." (col. 13, lines 52-53)

All of this takes place in a sheltered software development environment, ***before the programmer begins any actual coding.*** This is emphasized at two points in the *Desgrousilliers, et al.* patent's Summary of the Invention and also at both the start and at the end of the patent's Detailed Description:

> From a process standpoint, the invention comprises a method of preparing a suite of test scripts for testing a proposed subsystem control facility [SCF] set of requirements in a distributed system network ***prior to coding*** the proposed subsystem control facility, ... [Summary, Col. 3, lines 52-56 – ***emphasis added***]
>
> From an apparatus standpoint, the invention comprises a system for preparing a suite of test scripts for testing a proposed subsystem control facility [SCF] set of requirements ***prior to coding*** the proposed [SCF] design, ... [Summary, Col. 4, lines 15-18 – ***emphasis added***]
>
> Turning now to the drawings, FIG. 1 illustrates a knowledgebase developmental system 10 used to generate test scripts for use in testing a proposed subsystem control facility [SCF] set of requirements ***prior to preparing actual software cod***e for implementing the subsystem control facility [SCF]. [Start of Detailed Description, Col. 5, lines 27-31 – ***emphasis added***]
>
> As well now be apparent, the invention provides a comprehensive and consistent testing procedure which enables the selective preparation of a suite of test scripts tailored to a specific proposed subsystem set of requirements, which enables the developer/user to test the subsystem set of requirements ***without first coding the [subsystem control facility] SCF.*** [Next to final paragraph, Col. 16, line 66 to Col. 17, line 4 – ***emphasis added***].

Contrary to this, the present invention as claimed in claims 24 to 42 is directed towards a method of and an apparatus for auditing operating computers and their computer programs while they are actually operating in the field. The present invention is thus not a software development tool. The present invention is a tool for auditing the operations of computers and computer programs while they operate normally. Accordingly, the teachings of the *Desgrousilliers, et al.* patent are not at all relevant to the present invention. For this reason, the claims are believed to be allowable over the combination of *Weber* and *Desgrousilliers, et al.*

E. The *Weber* Patent Is Also Not Relevant To The Present Invention. It Relates To A Different Area of Technology – Computerized Secure Payment Transaction Clearance Methods and Systems

The *Weber* patent teaches the design of a distributed, Internet-based, merchant-to-customer secure payments transaction clearance method and system that utilizes such things as public-private encryption keys (col. 16, lines 22-25), Netscape's Secure Sockets Layer (SSL) protocol, and the Visa and-MasterCard Secure Electronic Transactions (SET) protocol to achieve an "on-line" cashless transaction clearing system which supports Internet shopping (col. 13, lines 4-18).

Briefly summarized and with reference to Figure 1B (and also to col. 13, lines 4-18), the *Weber* system works as follows: A customer's computer system 120 communicates over the Internet in a customer-merchant session (or linkage)[1] 150 (secured by the SSL protocol) with a merchant's computer system 130 to pay for goods or services (for the details of this interaction, see Figure 28 and col. 138, line 14 to col. 139, line 48). To verify the customer's ability to pay, the merchant's computer system 130 communicates over the Internet in a customer-institution session (or linkage) 170[2] (secured by the SET protocol) with a payment gateway 140, first to request (see Figure 4 and col. 16, lines 3-20) and then to receive back (see Figures 6A and 6B and col. 17 lines 20-27) authorization to accept the customer's payment, and then after accepting the payment, again to request the capture of the payment (see Figure 9 and col. 20, lines 10-22). The remainder of the patent simply explains in detail how this is done.

The *Weber* patent thus teaches what hardware and what communications protocols are needed to enable a customer, a merchant, and a bank or other payment gateway to transfer funds from a customer to a merchant with bank approval. In contrast to this, the present application and its claims describe a system that runs in the background and monitors many different types of operating computer systems throughout an enterprise, collecting and analyzing hardware and software configuration information and then reporting issues that require the attention of system management. These two fields of technology are unrelated.

---

[1] The *Weber* patent is inconsistent, speaking of the "customer-merchant session 150" and "customer-institution session 170" in col. 13, lines 4-18, and speaking of the "linkages 150 and 170" in col. 63, lines 54-57.

[2] See footnote 1 page 26.

For this reason, the claims are believed to be allowable over the combination of *Weber* and *Desgrousilliers, et al.*

F. Because They Relate to Different Fields of Technology, There Is No Motivation for One of Ordinary Skill in the Art to Combine the Teaching of the *Desgrousilliers, et al.* and *Weber* Patents

The *Weber* patent teaches how one may use computers to make payments securely over the Internet. It contains a brief passage which says that a merchant system may be taken out of service and tested while communicating not with the payment gateway of a bank but with a test payment gateway provided by the manufacturer of the merchant system.

The *Desgrousilliers, et al.* patent teaches how one can develop and test a set of requirements for a network management application program using test data before the program is actually written or coded.

With all due respect, applicants see nothing in either of these two patents that would motivate a computer scientist, after studying the teachings of both of these patents, to combine any of their teachings. The *Desgrousilliers, et al.* patent teaches only how the requirements for a program can be tested against test data long before the program is even written, and contrary to this the *Weber* patent teaches only how a normally operating merchant's computer system containing programs written long ago and actually operating can be tested as to the operating computer system's ability to handshake and communicate with a test or dummy bank computer that is substituted for the normal payment gateway computer of an actual bank. These teachings are not related in any way and are not combinable.

The Examiner maintains, as to representative independent claim 24 (final office action mailed on 6/3/2004, page 5, lines 8-20) that there is motivation to combine these two references (in the passage quoted from the Office Action and set forth below, all of the material cited by the Examiner has been inserted at appropriate points for clarity):

> Motivation – The portions of the claimed invention would have been a highly desirable feature in this art for
>
> - Early correction of errors (*Desgrousilliers et al,* Abstract,
>
> "A method and system for preparing a suite of test scripts for testing a proposed network management application. The proposed network

management application, termed a subsystem control facility (SCF), is first defined as a set of requirements with the aid of a developmental tool incorporating a subsystem knowledgebase and a test generation knowledgebase. The subsystem knowledgebase contains the rules governing the operation of a given network and a library of permitted commands, objects, attributes, modifiers and other data. The test generation knowledgebase includes information relating to those commands and objects specific to the proposed subsystem control facility set of requirements. A user interface coupled to the knowledgebases permits the selection of types of tests and specific commands and objects to be tested. Once test selection has been specified, test scripts corresponding to the selected tests are generated from the first and second knowledgebases for use in testing the proposed set of requirements for the SCF prior to coding the SCF. The suite of test scripts can be used to test the proposed SCF set of requirements, detect nonconformance with the rules and permit modification of the proposed set of SCF requirements. If necessary, additional testing may be performed using the same test scripts or a subset thereof, until the test results indicate no errors and a minimum number of warnings. Thereafter, the SCF product module can be coded. Errors in the developmental stage can be thus corrected early in the developmental cycle.")

- Convenient and secure information exchange (*Weber*, column 2, lines 21-55,

"To implement an automated, convenient transaction that can dispense some form of economic value, there has been a trend towards off-line payments. For example, numerous ideas have been proposed for some form of "electronic money" that can be used in cashless payment transactions as alternatives to the traditional currency and check types of payment systems. See U.S. Pat. No. 4,977,595, entitled 'METHOD AND APPARATUS FOR IMPLEMENTING ELECTRONIC CASH,' and U.S. Pat. No. 4,305,059, entitled 'MODULAR FUNDS TRANSFER SYSTEM.'

"The more well known techniques include magnetic stripe cards purchased for a given amount and from which a prepaid value can be deducted for specific purposes. Upon exhaustion of the economic value, the cards are thrown away. Other examples include memory cards or so called smart cards which are capable of repetitively storing information representing value that is likewise deducted for specific purposes.

"It is desirable for a computer operated under the control of a merchant to obtain information offered by a customer and transmitted by a computer operating under the control of the customer over a publicly accessible packet-switched network (e.g., the Internet) to the computer operating under the control of the merchant, without risking the exposure of the information to interception by third parties that have access to the network, and to assure that the information is from an authentic source. It is further desirable for the merchant to transmit information, including a subset of the information provided by the customer, over such a network to a payment gateway computer system that is designated, by a bank or other financial institution that has the responsibility of providing payment on behalf of the customer, to

> authorize a commercial transaction on behalf of such a financial institution, without the risk of exposing that information to interception by third parties.")
>
> Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made, to combine *Weber* with *Desgrousilliers, et al.* to obtain the invention of claim 24, a computerized method for auditing the software or hardware configurations of a plurality of computers in one or more enterprises. The modification would have been obvious because one of ordinary skill in the art would have been motivated to quickly and conveniently detect as well as securely report errors within enterprise computer systems.

The Examiner's position is thus that the present invention involves somehow combining the concept of correcting errors early, as illustrated by the *Desgrousilliers, et al.* patent's teaching that a "set of requirements" for a program should be thoroughly tested before the program itself is written, with the concept of providing convenient and secure information exchange, as is illustrated by the *Weber* et al. patent's teachings that sensitive customer payment transactions should be sent from customer to merchant to bank over the Internet in a secure manner.

The problem with the Examiner's position as stated above is that the present invention, as described in the application and as claimed in the claims, does not require nor teach these two concepts. It does not require nor teach that the requirements for a program should be thoroughly tested before the program is written; and it also does not require nor teach that all information sent over the Internet should be sent in a secure manner. Briefly summarized, the present invention (as defined in claim 24) does require and teach that hardware or software configuration information (or both) is to be collected from operating computers and is then to be analyzed by harnessing a list of specific analyzers, each containing executable program steps, to configuration data collected from a list of specific computers, the lists of analyzers and computers thus defining an audit task. The claims do not mention "early correction of errors," and the claims do not mention "convenient and secure information exchange." Accordingly, the two concepts listed by the Examiner and the two passages cited by the Examiner as illustrating these concepts and as demonstrating "motivation" are simply not relevant to the present invention as claimed.

Additionally, even if these two concepts and these two passages were relevant, they relate to entirely different topics of technology. Detecting and correcting errors earlier, rather

than later, is an entirely different concept that is unrelated to the concept providing a convenient and secure information exchange.

For both of these reasons, and with all due respect, the Examiner has not demonstrated any reasons why one skilled in the art to which the present invention pertains, mainly monitoring operating computers operating in the field, would be motivated to combine the teachings of a first patent's teachings on the testing of software requirements before the software is written with a second patent's teachings on how payments may be approved and made over the Internet. Hence, the claims are patentable over the combination of *Weber* and *Desgrousilliers, et al.*

G. Testing Procedures Taught in the *Weber* and *Desgrousilliers, et al.* Patents Are Carried Out Only On Computers and Software That Are Not Operating Normally in the Field

There is only brief mention in the *Weber* patent of system testing. The patent's abstract, summary, and claims are all directed to this relatively minor aspect of the patent's overall disclosure.

Weber's testing of the merchant's computer system 130 (Figure 1B) is not done in the manner taught by the present application. During *Weber*'s testing, the merchant's computer system 130 is taken entirely out of service and cannot be used by customers wishing to make actual payments to the merchant. Contrary to this, the present invention audits computer systems unobtrusively from the background while the computer systems are functioning normally in the field. *Weber*'s teachings in regard to testing are thus not relevant to the claims before the Examiner.

The merchant system testing aspect of the *Weber* patent's teachings is presented in the following passage, which refers to the merchant's computer system by its trademark, "vPOS":

> A bank distributes vPOS [a trademark name for the merchant system] via different sales channels. The first is direct from a bank to an existing merchant with whom the bank already has an existing relationship. In this case, a version of vPOS already customized for a bank is sent to the merchant, either directly by a bank, or through a third-party distributor or service bureau. ...

> The more interesting case ... is where the potential merchant acquires, through some non-bank channel, a "generic" vPOS which has not yet been customized to interact with a specific bank. This vPOS can communicate with a "test gateway", which the merchant may use to experiment with the various features of vPOS and to test the integration of the vPOS into a total online storefront. ...
>
> In order to actually transact business over the Internet, the merchant must first obtain a merchant ID from the merchant['s] bank with which he signs an acquiring agreement. For online payment processing, the merchant must also obtain an appropriate set of digital credentials in the form of public key certificates and possibly additional passwords, depending on the financial institution. Once these credentials are obtained, the merchant is ready to customize the already-obtained vPOS to communicate with a merchant bank's gateway.
>
> Using the built-in "serial number" certificate and the Test Gateway public key certificate (which is "hard-wired" into the vPOS software), it is possible to securely download a particular bank's customization applications to a specific copy of the vPOS software. Once the vPOS is appropriately configured, the last stage of customization download is to configure the vPOS so that it only responds to a public key certificate of the merchant's acquirer [bank]. This process is illustrated here in the context of a merchant who obtains a vPOS that talks to the VeriFone test gateway, and desires to customize the vPOS to interact with a gateway at a bank. [col. 61, line 28 to col. 62, line 2]

As this passage makes clear, this "off-line" testing procedure is not at all similar to the "on line" computer system auditing procedure described and claimed in the present application. For this reason, the claims are believed to be allowable over the combination of *Weber* and *Desgrousilliers, et al.*

The present invention, as defined by all the claims (see "Summary of Claim Subject Matter" presented above), is a system or method for producing an audit report identifying issues of concern to management arising in computers that are operating normally in the field. The PCs are being used by users for word processing, Internet exploring, and any number of other uses. The servers are running a wide variety of applications, accepting requests from client computers and responding, all in the normal manner. Meanwhile, configuration information is being gathered and analyzed and audit reports listing issues are being generated, all in the background, without the computers having to be taken out of service.

Contrast this with the teachings of the *Weber* patent, where a merchant's system for accepting and processing payments received from customers must be taken out of service –

disconnected from the bank's payment gateway host and connected instead to a test version of the payment gateway – so that testing can be carried out. Contrast this with the teachings of the Desgrousilliers patent, which teaches testing only of a program's requirements and not testing of the actual program itself, much less testing of the program while it operates normally.

Because neither the *Weber* nor the *Desgrousilliers, et al.* patent teaches gathering data from computers while they are operating normally and then analyzing that data, neither is relevant to any of the claims before the Examiner. For this reason, the claims are patentable over the combination of *Weber* and *Desgrousilliers, et al.*

H. The Claims, Considered Individually Element by Element, Are Patentable Over The Combination of *Weber* and *Desgrousilliers, et al.*

The Examiner, in the final rejection dated 6/3/2004, has cited 26 different textual passages and drawings found in the *Weber* and *Desgrousilliers et al.* patents against the individual elements of claims 24, 27, and 28. There is no explanation in the final rejection of how or why these textual passages and drawings teach the limitations found in the claim elements against which they are cited.

In the remainder of this brief, applicants will first present and explain each of these 26 cited passages and drawings, and then applicant will respond fully to each and every one of these objections to the patentability of these three claims.

In the discussion that follows, the elements of each claim are referred to as the "preamble," the "first element," the "second element," and so on. In the actual claims reproduced in the claim appendix, paragraph breaks within each claim separate the claim's "preamble" from its "first element," separate its "first element" from its "second element," and so on. For example, the "second element" of claim 24 is the passage "providing ... computer:" which begins and ends with a paragraph break and which is preceded by the claim's preamble ("A computerized ... steps of:") and by the claim's first element ("in a fully ... both;").

1. Group 1 Claims –Independent Claim 24 is Patentable Over The Combination of *Weber* and *Desgrousilliers, et al.* (Claims 25, 31-32, 38, and 40 Are Also Patentable for the Same Reasons)

All the reasons presented below supporting the patentability of claim 24 are equally applicable to the method claim 25, the apparatus claims 31-32, and the "means for" apparatus claims 38 and 40.

Briefly summarized, the present invention as defined in representative claim 24, requires and teaches that hardware or software configuration information (or both) is to be collected from operating computers and is then to be analyzed by harnessing a list of specific analyzers, each containing executable program steps, to configuration data collected from a list of specific computers, the lists of analyzers and computers thus defining an audit task. The analyzer's executable program steps compute and then present reports identifying significant issues, and these reports are then utilized by management for audit purposes.

The elements of claim 24 will now be discussed individually. The discussion presented here of independent claim 24's elements is equally applicable to all of the claims on appeal.

a) The First Element of Claim 24 Does Not Read Upon *Weber*

The Examiner indicates that the first element of claim 24, when modified as indicated below:

> in a <u>largely</u> ~~fully~~ automated manner, collecting from each of a plurality of networked computers configuration information defining each computer's software configuration or hardware configuration or both;[3]

reads upon the following passage -- the *Weber* patent's Abstract:

ABSTRACT

[3] For discussion purposes only, the Examiner modified this claim language, replacing "fully" with "largely" -- see page 3, line 6, of the Final Office Action mailed on 6/3/2004, and compare the Examiner's wording of this passage with the first element of claim 24 as set forth on page 84 in the claim appendix. The Examiner, at a later point in the Final Office Action (page 4, lines 4-9, particularly line 9), addresses the phrase "in a fully automated manner" in the context of the *Desgrousilliers, et al.* patent. A response to this is presented beginning on page 56 of this brief.

> An architecture for verifying the operation of a remote transaction clearance system is disclosed. A merchant-controlled computer communicates with a test gateway computer over a communications channel. The test gateway computer responds with simulated transaction responses. In another aspect of the invention, the transaction responses include configuration data that is used by the merchant-operated computer to configure itself to access a production gateway computer.

The second and third sentences in this passage describe a merchant's computer system that can be tested by sending test payment transaction information not to a bank's host computer but to a "test gateway" which responds as if it were a bank's host computer -- but the test gateway simply sends back dummy responses. The data being transferred here is not hardware or software "configuration information," as is required by the first element of claim 24. The data is dummy or test payment transactions. Clearly, this first element of claim 24 clearly does not read upon the first three sentences in this passage.

The final sentence of this passage speaks of sending configuration data ***to*** the merchant's computer, not of collecting configuration information ***from*** the merchant's computer. This is done to reconfigure the merchant's computer out of its "default" configuration, in which it only communicates payment information to a test host computer, and into the configuration that is required by a particular bank's host computer so that the merchant's computer system may thereafter communicate payment information to that particular bank's host computer. This final sentence does mention configuration data, but the configuration data mentioned is flowing ***towards*** the merchant's computer to reconfigure that computer -- it is not being ***collected from*** the merchant's computer, as the claim language requires. Because the first element of claim 24 specifically calls for the "collection" of "configuration information", and not the dissemination of such information to reconfigure a computer in the field, the first element of claim 24 clearly does not read upon this final sentence.

The Examiner further indicates that the first element of claim 24 also reads upon the following passage taken from the *Weber* patent:

> The unique architecture of the Cardholder 120, Merchant 130 and Gateway 140, as shown in FIG. 1B, provides communication capability between the modules utilizing the Internet to support [sessions or] linkages

> 150 and 170. Since the Internet is so pervasive, and access is available from virtually any computer, utilizing the Internet as the communication backbone for connecting the cardholder, merchant and access to the authorizing bank through a gateway allows the merchant vPOS software to be remotely located from the merchant's premises. [col. 63, lines 54-63]

The sessions (or linkages[4]) 150 and 170 discussed here are each defined by handshake protocols. The session (or linkage) 150 (discussed above on pages 25-26) is a secure Internet handshake that takes place between the "cardholder" or customer's system 120 and the merchant's system 130 to implement an "I want to pay for a purchase" handshake. (This handshake protocol is fully described in Figure 28 of the *Weber* patent.) The session (or linkage) 150 is thus not a session that collects "configuration information defining each computer's software configuration or hardware configuration or both" as is required by the first element of claim 24. It is, instead, a session that enables the customer to offer to pay, and it enables the merchant to accept payment. The claim language clearly does not read upon this passage with respect to the session (or linkage) 150.

The session (or linkage) 170 (also discussed above on pages 25-26) is a secure Internet handshake session that takes place between the merchant's system 130 and the payment gateway 140 to implement a "can I accept payment from this customer" handshake session. (The handshake protocol is fully described in Figure 4, Figures 6A and 6B, and Figures 12A and 12B of the *Weber* patent.) The session (or linkage) 170 is thus also not a session that collects "configuration information defining each computer's software configuration or hardware configuration or both" as is required by the first element of claim 24. It is, instead, a session that enables the merchant to obtain permission from the bank (the payment gateway) to accept payment and to later request "capture" of the payment. The claim language clearly does not read upon this passage with respect to the session (or linkage) 170.

The Examiner also indicates that the first element of claim 24 reads upon the following additional, rather lengthy, passage taken from the *Weber* patent:

---

[4] The *Weber* patent is inconsistent, speaking of the "customer-merchant session 150" and "customer-institution session 170" in col. 13, lines 4-18, and speaking of the "linkages 150 and 170" in col. 63, lines 54-57.

> A Data Manager provides storage and retrieval of generic data items and database records. It is assumed that data fields, index fields or entire data records can be marked as encrypted and the encryption process is largely automated. The data manager has no specific knowledge of database records appropriate to different payment methods. This layer is separated out so as to reduce changes required when new payment methods are introduced. However RSA key pairs and certificates might be considered as "simple" data types. This component also provides an abstraction which supports wallet files on computer disk or contained in smart cards.
>
> The Open Data Base Connectivity (ODBC)/Java Data Base Connectivity (JDBC) component provides Data Base Connectivity where formal database components are required. An embodiment of the Smart Card Wallet allows wallet data to be stored and/or secured by a cryptographic token.
>
> A preferred embodiment includes a single file or directory of files comprising a "wallet" which contains personal information and information about multiple payment methods with the preferred implementation. These payment methods (Visa cards, debit cards, smart cards, micro-payments etc.) also contain information such as account numbers, certificates, key pairs, expiration dates etc. The wallet is envisaged to also contain all the receipts and transaction records pertaining to every payment made using the wallet. A Cryptographic API component provides a standard interface for RSA and related cryptographic software or hardware. This support includes encryption, signature, and key generation. Choice of key exchange algorithm, symmetric encryption algorithm, and signature algorithm should all be configurable. A base class stipulates generic behavior, derived classes handle various semantic options (e.g. software based cryptography versus hardware based cryptography.) [col. 137, line 26-63]

This passage is pulled from a discussion of "Certificate Processing," where the point is made that the merchant needs to obtain reassurance that the "cardholder" has a valid card, while the "cardholder" needs reassurance that the "merchant" is authorized by the bank to accept payment. This particular passage discusses various aspects of a "wallet" feature, where the "wallet" is a secure data set that contains credit card numbers and other confidential personal information of the "cardholder." This passage also discusses security techniques for managing such information.

But this passage contains no mention of the "collecting" of "configuration information defining each computer's software configuration or hardware configuration" as is

required by the first element of claim 24. The claim language clearly does not read upon this passage.

Accordingly, the first element of claim 24 does not read upon any of the passages in the *Weber* patent which the Examiner has pointed out.

b) The Second Element of Claim 24 Does Not Read Upon *Weber*

The Examiner indicates that the second element of claim 24:

> providing a plurality of analyzers ~~based on expert knowledge~~[5], each analyzer comprising the executable program steps needed to compute, from selected configuration information gathered from a single computer, a report identifying at least one issue relating to the computer;

reads upon the last sentence in the following passage of the *Weber* patent (the entire passage is reproduced, rather than just the last sentence cited by the Examiner against claim 24, because the Examiner later cited this entire passage against the independent apparatus claims 31 and 38):

> Transaction Performance Monitoring and Measurement
>
> The "hits" performance indicators are available from the Web server. Statistics can be generated at any time to highlight the load pattern or to pinpoint the time when the server was most active.
>
> Gateway statistics about transaction requests (by transaction type) and transaction results (e.g., success, failed due to host, failed due to authentication, etc.) can be determined at any time for a particular time interval by generating a report. [col. 124, lines 9-18]

This passage is taken from a later section of the *Weber* patent that describes the front end portion of the payment gateway 140 (Figure 2). The payment gateway 140's front end is called an "Internet transaction gateway." This "gateway" un-encrypts and transforms

---

[5] For discussion purposes, the Examiner left the phrase "based on expert knowledge" out of this claim element when reproducing the language of this claim element in the rejection (Final Office Action mailed on 6/3/2004, page 3, line 12) to indicate that the Examiner did not find anything in the *Weber* patent that anticipated this particular phrase. The phrase is shown here stricken through. The Examiner, at a later point in the Final Office Action (page 4, lines 4 to 8 and lines 10 to 17, particularly lines 10 to 17), addresses this phrase in the context of the *Desgrousilliers, et al.* patent. A response to this is presented beginning on page 57 of this brief.

transactions received from merchant systems 130 over the Internet into a different format, a format which an accepting bank's "legacy processor" host computer can accept – a transaction format essentially identical to what is generated by today's merchant credit card scanner boxes and delivered by telephone to such a legacy processor bank host computer. (See col. 119, lines 13-42 – the "Internet transaction gateway" appears in Figure 21.)

This Internet transaction gateway "provides logging, monitoring, reporting, and system administration" functions. (col. 120, line 19 and lines 35-36) It includes a transaction logger 2150 (Figure 21) which passes records via a database server 2160 to capture transaction logs in a database 2180. These transaction logs contain data concerning each transaction, where a transaction is a payment request from a merchant's computer system sent to an accepting bank's host computer, a payment capture request from a merchant's computer system sent to an accepting bank's host computer, or a response to such requests sent back to the merchant's computer system from the accepting back's host computer. These transactions contain information such as the identity of the customer, the credit card number, the amount of the transaction, etc. These transactions do not contain configuration information defining some computer's software configuration or hardware configuration, as the first element of claim 24 requires.

The above passage pointed out by the Examiner simply says that this Internet transaction gateway also includes some type of computer program which, when provided by an operator with a time range, can generate for that operator statistics concerning the percentage of payment request transactions that were honored and the percentage of payment transactions that were not honored for various reasons.

This Internet transaction gateway computer program does not comply with specific requirements set forth in the second element of claim 24: This computer program does not "compute, from selected configuration information gathered from a single computer, a report identifying at least one issue relating to the computer". First, this program cannot generate such a report because this program does not receive as input data any software or hardware configuration information – it receives as input data only payment transaction information. Secondly, this program does not compute a report identifying issues relating to a specific computer – it computes reports relating to customer-merchant payment transactions and their

successful or unsuccessful processing. Accordingly, the claim language clearly does not read upon this passage.

The Examiner further indicates that the second element of claim 24 also reads upon the following passage taken from the *Weber* patent:

> Analyze SetRequest
>
> FIGS. 52A and 52B describe the AnalyzeSetRequest routine. This routine is by Step 5110 as illustrated in FIG. 51. Execution begins in Step 5200. In Step 5205 the various fields in the SET record are obtained, as will be more fully disclosed below with respect to FIGS. 56A and 56B. In Step 5210 the Gateway checks the retry count. A retry count is zero indicates that the request being analyzed is a new request, and control proceeds to Step 5212, indicating a new request. If the retry account is non-zero, this means that the request is a retry of a prior request, and control proceeds to Step 5214 where a retry is indicated.
>
> Following either step 5212 or 5214, execution proceeds to Step 5215. In Step 5215 the Gateway checks to see whether the request represents a "stale request," as will be more fully described below with respect to FIG. 57. In Step 5220, the Gateway tests the result of the stale check from Step 5215. If the request is stale it is marked as stale in Step 5222. Otherwise the record is marked as not stale in Step 5224. Following either Step 5222 or Step 5224, control proceeds to Step 5230. In Step 5230 a message representing the SET request is inserted into the database for tracking purposes, and control proceeds to Step 5240.
>
> In Step 5240 the Gateway checks to see if the request had been marked stale in Step 5222. If so, it proceeds to Step 5242, exiting with an error condition. In Step 5245, the Gateway attempts to retrieve from the database a message corresponding to the current SET request, ... [col. 142, lines 32-59]

To understand this passage, one must first understand the meaning of the word "SET" and the phrase "SETRequest" both of which appear this passage.

"SET" means the "Secure Electronic Transfer" protocol adopted by Visa and MasterCard. (col. 2, lines 58-64) Referring to Figure 3, the *Weber* patent teaches that the "... session 170 [which occurs between the merchant's computer system 130 and the payment gateway computer system 140] operates under a variant of a secure payment technology such as the SET protocol ...." (col. 13, lines 14-18)

The patent notes that all the transactions originate with the merchant's computer system 130, and not with the gateway computer system 140 (col. 13, lines 14-18). Accordingly, it is appropriate to think of every transaction as being a "request" made to the gateway computer system 140. From this it can be seen that a "SET request" is simply a request (or transaction) initiated by a merchant's computer system 130, formulated using the Visa-MasterCard "SET" protocol, and sent to the payment gateway computer system 140 which passes it on to the host computer system of the merchant's acquiring bank.[6]

The full title of the above passage, "*AnalyzeSetRe*quest," is the name of a computer program or routine. The above passage is the first part of an explanation of what this program or routine does. Before this passage can be fully understood, the environment within which this program or routine operates needs to be explained. This environment will be described in the next three paragraphs, and then an explanation of this program will be presented.

It will be recalled from discussions presented above that the gateway computer system 140 includes an Internet Transaction Gateway (shown in Figure 21) and the bank's host computer system (not shown). All requests and all replies to requests pass through this Internet Transaction Gateway when traveling between the merchant's computer system 130 and the bank's host computer system. Transmissions to and from the merchant's computer system enter and leave this Gateway at a secure server 2102 (Figure 21) (which connects to the public Internet), and transmissions to and from the bank's host computer system enter and leave this Gateway at a socket TCP multiplexer 2130 (Figure 21) (which connects to a private network link 2140 leading to the bank's host computer system). Decryption of all requests received from the merchant's computer system 130 and encryption of all replies sent back to the merchant's computer system 130 take place at 2120 (Figure 21) within this Gateway.

The remaining steps carried out on requests and replies by this Gateway are shown in Figure 51. Decrypted requests received from a merchant's computer system 130 are

---

[6] As used in the *Weber* patent, the term "request" (judged from the context of usage within the *Weber* patent) normally includes by implication both an initial request message sent out by the merchant's computer system and also any related reply message made in response to this request by the bank's host computer system. Thus, the request message and any reply message appear to be included in the term "request" as used here.

translated into the Bank's format (the "forward translate" step 5135) and are then sent on to the bank's host computer system (the "host communication" step 5145). When the bank's host computer system provides a reply, that reply is translated back into the merchant computer system's format (the "reverse translate" step 5155) and is then encrypted and sent back (the "proceed to respond" step 5120) to the merchant's computer system 130.

Figure 51 illustrates and includes a program "looping" process, where each request and its associated reply (if any) is processed by at least four repetitive passes around the program control loop defined by the four test steps 5120, 5130, 5140, and 5150, taking a different branch 5135 (forward translate), 5145 (host communication), 5155 (reverse translate), and 5120 (proceed to respond) during each pass around this loop to complete all the necessary tasks.

The AnalyseSetRequest program, described in the above passage that was cited by the Examiner, appears at step 5110 in this same Figure 51. The figure is not drawn correctly – the program control loop just described should pass through this *AnalyzeSetRe*quest program 5110, and not bypass it. During every pass around this loop, the AnalyseSetRequest program 5110 is called upon to examine or analyzes the request and any associated reply, and then the *AnalyzeSetRe*quest program 5110 sets the necessary software switches to control which of the steps 5135 (forward translate), 5145 (host communication), 5155 (reverse translate), and 5120 (proceed to respond) the request or reply is subjected to during the next pass around the program control loop shown in Figure 51.

The above passage cited by the Examiner is thus a description of a computer program that periodically analyzes data associated a merchant computer system's requests, and possibly also with a corresponding bank host computer system's replies to these requests. This program then makes a decision based on this analysis as to what processing these requests (and sometimes also these replies) are to be subjected to next. Should the next step performed be forward translation of the request from the merchant computer system's data format into the bank host computer system's data format? Should the next step performed be sending the translated request on to the bank's host computer system? Should nothing be done with this request, since we are awaiting a reply from the bank's host computer system? Has excessive time elapsed, so that an error should be declared and the processing of this

request and any reply should be aborted? This is the type of analysis that the program "Analyze SetRequest" performs, and this is what is described in the above passage which the Examiner maintains the second element of claim 24 reads upon. (For further details, see the overview flowchart of this *AnalyzeSetRe*quest program that is presented in Figures 52A and 52B and the accompanying text of the patent, col. 142, line 32 to col. 144, line 31. An actual program listing appears in columns 144 and 145.)

This passage cited by the Examiner does describe a computer program 5110 containing executable program steps, and this program 5110 does perform some form of analysis. But this program 5110 does not receive as its input data "selected configuration information gathered from a single computer" as the second element of claim 24 requires, and it also does not "compute" from such information "a report identifying at least one issue relating to the computer" from which the configuration information was gathered. In addition, and with reference to the first element of claim 24, the "configuration information gathered" must define the "computer's software configuration or hardware configuration or both." But the only data gathered by the Internet Transaction Gateway and subjected to analysis by its *AnalyzeSetRe*quest program 5110 is data defining the status of requests to authorize a merchant to receive a payment from a customer and a subsequent request to capture that payment and the like, as has been explained.

The claim language of the second element of claim 24 clearly does not read upon this passage in any way. Accordingly, the second element of claim 24 does not read upon any of the passages cited by the Examiner.

c) Third Element of Claim 24 Does Not Read Upon *Weber*

The third element of claim 24 is a method step which calls for:

> defining at least one task definition comprising a list of one or more of said computers and a list of one or more of said analyzers;[7]

[7] For discussion purposes only, the Examiner, in his final rejection, presented this passage not as it is shown here, but in an altered form that completely omitted the two occurrences of the phrase "a list of" (see

This third element of claim 24 calls for a "task definition" that defines a specific auditing task. An auditing task is an order that is presented to an auditing computer system ordering it to examine configuration information gathered from a specific list of computers by processing that information using the executable program steps included in a specific list of analyzers. The task definition lists the computers which are to be audited, and it lists the analyzers whose executable program steps are to carry out this analysis. With reference to the present application, a task definition system 810 appears in Figure 8 of the present application. It includes an assessment task 814 that contains a list of nodes (which term includes computers) and a list of analyzers, as is shown in Figure 8.

The Examiner indicates that this third element of claim 24 reads upon the following passage taken from the *Weber* patent:

> ... The vPOS configuration setup function is used to setup the vPOS configuration data. The vPOS configuration data is divided into different tables, for example, the Card/Issuer Definition Table (CDT), the Host/Acquirer Definition Table (HDT), the Communications Parameters Table (CPT) and the Terminal Configuration Table (TCT). ... [col. 39, lines 13-19]

This passage is taken from three paragraphs (col. 38, line 59 to col. 39, line 25) of the *Weber* patent that list the various software functions a merchant can call upon to perform tasks relating to the merchant's computer system 130 (referred to in this passage by its trademark name "vPOS"). The "configuration setup function" discussed in this passage permits the merchant to adjust certain merchant computer system configuration information values that are stored within the merchant's computer system and that form a part of its software configuration. These configuration information values include:

> 1. A Card Issuer Definition Table (CDT), which contains information specific to each credit card issuer (Visa, MasterCard, etc.).[8]

---

page 3, lines 17-18 of the Final Office Action mailed on 6/3/2004, and compare to the third element of claim 24 set forth on page 2 of this reply). With all due respect, applicants object to this alteration of this claim language, even for discussion purposes, since revising this passage in this manner requires a "task definition" to contain "computers" and "analyzers" rather than simply to contain "a list of ... computers and a list of ... analyzers." Accordingly, this change clearly alters the substantive meaning of this entire passage in a way that changes the substantive scope of the claim and also renders this passage meaningless in the context of the present invention.

[8] For example: the card issuer's name; the minimum and maximum number of digits permitted in the credit card issuer's credit card numbers (called a "primary account number" or "PAN"); the low end and the

> 2. A Host/Acquirer Definition Table (HDT), which contains Information specific to the merchant's acquiring bank ("acquirer") and to that bank's host computer system ("host").[9]
>
> 3. A Communications Parameter Table (CPT), which contains additional Information also specific to the merchant's acquiring bank ("acquirer") and to that bank's host computer system ("host").[10]
>
> 4. A Terminal Configuration Table (TCT), which contains only two values: the merchant's name (col. 41, lines 7-8) and a "Lock Flag" that signals whether the merchant's computer system is "locked" or "unlocked." (col. 41, lines 1-10)

The passage cited by the Examiner does not read upon the third element of claim 24. The passage simply describes a computer program that is built into the merchant's computer system 130 and that permits the merchant to adjust configuration information values stored within the system 130. This passage does not speak of defining any "task definition", as the third element of claim 24 requires. The passage does not speak of including in such a task definition a "list of ... computers" which are to be analyzed or audited, as this third claim element also requires. This passage does not speak of including in such a task definition any "list of ... analyzers" (or any other computer programs) as this third element further requires, particularly not an analyzer whose "executable program steps" must be able to "compute ... a report" form the "configuration information gathered from a single computer, a report identifying at least one issue relating to the computer" as claim element 2 requires. This passage does not teach computing anything from the configuration information found on the merchant's computer system 130 – it only teaches assisting the merchant in altering such configuration information, and it does not teach computing reports from such information following some type of analysis. The only executable program steps mentioned in this passage is the "configuration setup function" whose purpose is not to test configuration information nor to compute anything from configuration information but simply to permit a

---

high end of the credit card number range for this credit card issuer; which types of transactions the card issuer permits; etc. (col. 40, lines 13-50)

[9] For example: the bank's merchant identifier; the terminal ID number assigned to the merchant by the bank; the name identifying the bank's host computer; the reference number to be used as part of the next transaction; etc. (col. 39, lines 20 to col. 40, line 12)

[10] For example: the Primary Host Address (telephone number or IP address, etc.); the Secondary Host Address (telephone number or IP address, etc.) to be used as a backup; a timeout value – how quickly the merchant's terminal should receive a response from the bank's host computer; etc. (col. 40, lines 50-67)

human to adjust configuration information. This claim language clearly does not read upon this passage.

The Examiner further indicates that this third element of claim 24 reads upon the following passage taken from the *Weber* patent:

> Web Servers can process many transactions at a time, so payment requests can often occur simultaneously. Thus, the vPOS Software must have support for multi-tasking and provide support for multiple threads to be active at the same time in the same system as well as the same process. This requirement is relatively straight forward. However, the authorizing banks require that all transaction requests include a Terminal ID (TID), and, for many banks, no single TID may be active in any two transaction requests that overlap in time. Thus, the vPOS requires dynamic allocation of TIDs to requesting threads. (col. 63, lines 14-24)

This passage explains that bank host computers conventionally assign one terminal ID to each merchant and do not permit two transactions to take place from the same merchant at the same time. Such an arrangement would not permit several Internet payments to be approved simultaneously. The answer proposed here is that the merchant's computer system 130 can be given a set of several terminal IDs, and then, through the use of conventional "multithreading" or multi-tasking techniques, it can process simultaneously as many payment transactions as the computer system 130 has terminal IDs assigned to it by the bank.

The third element of claim 24 does not read upon this passage. There is nothing in this passage that speaks of defining a task definition by listing analyzers that are to be exercised and by listing the computers on whose configuration data those analyzers are to perform computations. This claim language clearly does not read upon this passage.

The Examiner finally indicates that this third element of claim 24 also reads upon the following passage taken from the *Weber* patent:

> vPOS Multi-Merchant Processing
>
> Multiple merchant processing refers to the ability of a plurality of merchants to process their individual vPOS transactions securely on a single computer. The architecture relies on each payment page obtaining the merchant name in a hidden field on the payment page. The vPOS engine receives the merchant name with a particular transaction and synchronizes the

> processing utilizing a Set Merchant method. This command causes the vPOS API to look up a unique registry tree based on the merchant name. This process causes the vPOS engine to engage the appropriate configuration to process the transaction at hand utilizing a Registry Tree. A registry tree contains Card Definition Tables (CDT)s, Acquirer Definition Tables (ADT)s, Merchant Definition Tables (MDT)s, Protocol Configuration Tables (PCT)s, etc. The CDTs point to specific ADTs since each supported card can be supplied by a distinct acquirer. This is one form of split connection. Each of the ADTs in turn point to PCTs, and some acquirers can support multiple parallel gateways. A merchant's name refers to a unique database in the database management system which contains for example, TIDs.
>
> So, for example, to fully qualify a particular merchant in a multi-merchant system, the Acquirer Definition Table is queried to ascertain the particular Gateway (VFITest), then if Bank of America requires verification of network communication information, the particular CardDT is accessed with for example VISA. The particular merchant will service VISA transactions utilizing a particular acquirer. The particular piece of merchandise will also be detailed in a data base. Finally, the merchant Configurations will also be stored in the database to facilitate E-mail and name lookup. [col. 64, lines 15-45]

This passage says nothing more than that two or more merchants may share a single merchant's computer system 130, and it discusses how such a system keeps the bank payment information for the multiple merchants separately stored in separate tables. Since the present application does not have any teachings on whether or not multiple users may share a single computer, and since this topic does not come up in any claim element, this passage would appear to be irrelevant to the present invention. The third element of claim 24, which calls for a task definition that includes a list of analyzers that are to process configuration information gathered from a list of computers, clearly has not relevance to this passage. This claim language clearly does not read upon this passage.

Accordingly, the third element of claim 24 does not read upon any of the passages cited by the Examiner.

d) The Seventh Element of Claim 24 Does Not Read Upon *Weber*

The seventh element of claim 24 reads as follows:

> utilizing any issue identifying report generated during the processing step for audit purposes

This seventh element of claim 24 does not read upon any type of financial auditing activity or financial record management activity. In its preamble, claim 24 says specifically that the novel method described in the claim is "a computerized method for auditing the software or hardware configurations of ... computers". It is not a method for auditing financial and bank records. It is not a method for auditing records of inventory turnover (relating to sale of goods). It is not a method of auditing (credit card) payment transactions of a merchant. Consistent with this, the "reports identifying at least one issue relating to the computer" (claim 24, element 2) which result from computations performed upon "configuration information defining each computer's hardware configuration or software configuration or both" (claim 24, element 2), and which the sixth element of claim 24 requires to be utilized "for audit purposes," are clearly not financial nor inventory (of goods) audit reports. They are reports concerning the hardware and software configurations of specific computers. The phrase "utilizing any ... report ... for audit purposes" which appears in the sixth element of claim 24 is, accordingly, limited to reports useful for auditing the hardware and software configurations of computers. This claim language cannot be construed to cover the types of financial reports of a merchant that a certified public accountant would be called in to examine during a financial audit.

The Examiner indicates that this seventh element of claim 24 reads upon the following passage of the *Weber* patent:

> ... The merchant could utilize any other computer attached to the Internet utilizing a SSL or SET protocol to query the remote vPOS system and obtain capture information, payment administration information, inventory control information, audit information and process customer satisfaction information. ... (col. 64, lines 1-7)

This passage is taken from the middle of a longer passage (col. 63, line 52 to col. 64, line 11) which explains that the merchant's computer system 130 (referred to here by its trademark name "vPOS") does not have to be installed on the premises of the merchant's place of business. Instead, the merchant's vPOS computer system 130 may be located "at a central host site where merchant vPOS systems for various merchants all resided on a single host with their separate access points on the Internet." (col. 63, line 65 to col. 64, line 1). Then, as this passage says: "The merchant could ... query the remote vPOS [host] and

obtain" [col. 64, lines 1-7] from it certain information, all relating to the sales of goods and the receipt of payments, including the following:

> 1. "Capture" information [col. 64, line 4] -- in the context of a merchant's payment computer system, meaning information about requests for the "capture" of specific customer credit card payments which requests are sent to the host computer system of the merchant's acquiring bank, and whether those requests for "capture" were approved or not. (Figure 9 and col. 20, lines 23-39)
>
> 2. "Inventory control" information [col. 64, line 5] -- in the context of a merchant's payment computer system, meaning records of how many of what types of goods have been sold by the merchant's computer system to customers.
>
> 3. "Payment administration" information [col. 64, lines 4-5] -- in the context of a merchant's payments computer system, meaning records for each customer of payments made by that customer.
>
> 4. "Audit" information [col. 64, line 6] -- in the context of a merchant's payment computer system, meaning financial information about payments received and inventory transfer information about goods sold, all gathered and maintained specifically for the merchant's financial auditors to study at a later time.
>
> 5. "... And Process Customer satisfaction" information [col. 64, lines 6-7]. The *Weber* patent does not define what this information is nor how it might be processed.

This is all financial information – reports of customer payments received, reports of inventory turnover, and reports of payments captured for the merchant by the merchant's accepting bank. (Some of this information may also be reports of customer survey information gathered incidental to the sale and payment processing processes.) These are not reports calculated from hardware and software configuration information of the type that the sixth element of claim 24 requires be used for computer hardware and software audit purposes. For this reason, the claim language clearly does not read upon the above passage.

The Examiner further indicates that the seventh element of claim 24 also reads upon the following passage also taken from the *Weber* patent:

> Gateway statistics about transaction requests (by transaction type) and transaction results (e.g., success, failed due to host, failed due to authentication, etc.) can be determined at any time for a particular time interval by generating a report. [col. 124, lines 14-17]

This passage is taken from a description of the features of the payment gateway computer system 140 (Figure 1B), the one that the merchant's computer system 130 contacts to obtain authorization from the merchant's accepting bank to receive individual payments made by credit card (etc.) and that the merchant's computer system 130 later contacts to request capture of the individual payments once the corresponding purchase transaction is completed with the customer.

As was discussed above, this payment gateway computer system 140 includes an Internet Transaction Gateway (shown in Figure 21 – see col. 119, lines 13-42) which includes system administration software 2195 that, as the above passage indicates, can be adjusted by the merchant to generate statistical reports indicating, as a possible example, what percentage of customer credit card payment authorization requests were approved and what percentage were disapproved during the past 24 hour time interval.

This is not a report computed from collected configuration information defining the hardware or software configuration of a computer. This is a report of financial transactions and their success or failure. The claim language clearly does not read upon this passage.

Accordingly, the seventh element of claim 24 does not read upon the passage cited by the Examiner.

e) The "full automation" Limitation in the First Element of Claim 24 Does Not Read Upon Desgrousilliers et al., and Accordingly the First Element of Claim 24 does Not Read Upon The Combination of *Weber* with *Desgrousilliers, et al.*

Returning to the first element of claim 24 (discussed above with respect to the fact that it does not read upon the *Weber* patent):

> in a fully automated manner, collecting from each of a plurality of networked computers configuration information defining each computer's software configuration or hardware configuration or both;

It will be recalled that the Examiner indicated this claim language reads upon a passage in the *Weber* patent (this was discussed and refuted above), but the Examiner conceded that the *Weber* patent doesn't explicitly teach "full automation" of collecting configuration information, a requirement of this first element of claim 24. Accordingly, the Examiner has

based his rejection upon the combination of a passage in *Weber* (discussed above) with a passage in Desgrousilliers et al which is set forth here (three sentences are set forth here – the third sentence presented (col. 13, lines 60-62) is the passage pointed out by the Examiner as illustrating "full automation"):

> ... [T]he test results in storage device 29 can be manually analyzed by the user, by providing a printout of the contents of each command and response buffer. In fact, in the preferred embodiment, the SCF response format and attribute values must be manually reviewed. The SPI test results, on the other hand, are all automatically analyzed and the result is preferably printed out with comments. ... (col. 16, lines 56-62)

The problem with the Examiner's position is that the first element of claim 24 specifically calls for "collecting" "configuration information" from "a plurality of networked computers" in "a fully automated manner." Directly contrary to all of this, the passage above does not mention any "collecting" activity (the "test data" is all conveniently present in a single storage device 29 shown in Figure 8); and the passage above also does not mention hardware or software configuration information (instead it mentions test results resulting from tests of an unwritten program's requirements carried out by a "test SCF unit 25" shown in Figure 8). In addition, the process of analysis and report generation described is only partly automated – it is partly done manually (as the full passage quoted above clearly indicates). Add to this the fact that the *Desgrousilliers, et al.* patent's teachings are not at all relevant either to the present claims or to the *Weber* patent (as was explained above), and it can be seen that the claim language clearly does not read upon the final sentence in the passage presented above. Accordingly, the first element of claim 24 does not read upon this *Desgrousilliers, et al.* passage either alone or in combination with one of the *Weber* passages.

f) The Second Element of Claim 24 Does Not Read Upon The Combination of *Weber* with *Desgrousilliers, et al.*

The Examiner conceded that the italicized language appearing below in the second element of claim 24 (discussed above with respect to the fact that this claim language does not read upon the *Weber* patent):

> providing a plurality of analyzers ***based upon expert knowledge***, each analyzer comprising the executable program steps needed to compute, from

> selected configuration information gathered from a single computer, a report identifying at least one issue relating to the computer;

was not taught by the three *Weber* patent passages (discussed above) that the Examiner has cited against this claim element.

Accordingly, the Examiner cited the three *Weber* patent passages (summarized below), in combination with the passage below which is taken from the *Desgrousilliers, et al.* patent (the examiner did not cite the first two sentences presented below -- they are included here for clarity):

> A knowledgebase design tool ... permits a subsystem developer to reliably design a new subsystem control facility using a graphical interface with a knowledgebase. ... In this design tool, the subsystem designer is able to call up and examine in an orderly manner all of the specific commands and objects to be incorporated into the specific proposed subsystem set of requirements, from a generic set of permitted commands and objects. In addition, the subsystem designer is able to select appropriate error messages, and help and error texts, for display purposes. By individual selection using the interface keys and guided by the interface prompts, the subsystem designer can construct the proposed set of requirements. Improper selections by the developer are noted on the screen of the graphics interface, thereby enabling the developer to correct syntactical errors and rules errors throughout the design process. Once the developer has completed the initial set of requirements, the knowledgebase system then generates an appropriate external specification in text form which can be reviewed for compliance with the system rules and standards by the quality assurance and programmatic interface experts. In addition, the knowledgebase system also generates a set of help/error text files tailored to the specific commands and objects of the new set of requirements, a data dictionary language tailored also to the set of requirements and a binary table file which contains in binary form the command syntax, display format and programmatic interface buffer formatting information needed to match the new subsystem design.
>
> SUMMARY OF THE INVENTION
>
> The invention comprises a method and system incorporated into the above described developmental tool for providing automatic test generation after the developer has completed the set of requirements, which permits comprehensive testing of both positive and negative performance of the new set of requirements, and which permits a new proposed subsystem control facility to be tested thoroughly and the testing results analyzed prior to implementation in code form. The analyzed test results include traceability to the requirements as expressed in the developmental tool knowledgebase.

> From a process standpoint, the invention comprises a method of preparing a suite of test scripts for testing a proposed subsystem control facility set of requirements in a distributed systems network prior to coding the proposed subsystem control facility, ... [col. 3, lines 18-56]

The Examiner cited this passage in combination with three passages taken from the *Weber* patent (discussed fully above and briefly below) to demonstrate the presence in the prior art of analyzers comprising "executable program steps" that were "based on expert knowledge" and that could "compute, from selected configuration information gathered from a single computer, a report identifying at least one issue relating to the computer".

With all due respect, this passage from the *Desgrousilliers, et al.* patent teaches no such thing, alone or in combination with *Weber*. The Desgrousilliers et al. patent's teachings are directed to assisting a programmer who has yet to write a program in first defining the requirements of that proposed program and then in exercising those program requirements against test data to see if the requirements are acceptable. This is a programmer's tool for use in a software development laboratory, not a support engineer's tool that can collect configuration information from computers operating in the field, and then analyze that information.

What the Desgrousilliers et al. patent lacks in the above passage, as well as elsewhere, is any teachings concerning: "collecting" any information at all; collecting "in a fully automated manner;" collecting "from each of a plurality of networked computers;" collecting information that can be characterized as "configuration information;" and collecting configuration information "defining" any "computer's software configuration or hardware configuration or both." In other words, the teachings of the *Desgrousilliers, et al.* patent are so far removed from the present invention that they are entirely irrelevant to the second element of claim 24.

Likewise, the combination of this passage taken from the *Desgrousilliers, et al.* patent with the three passage of the *Weber* patent which the Examiner has cited is no more enlightening, and the second element of claim 24 still does not read upon such a combination.

The Examiner first cited the *Weber* patent's Abstract, which speaks of a merchant's computer system sending requests to approve payments to a "test" central bank computer that

takes the place of a real central bank computer for testing purposes. *Weber*'s abstract speaks about testing an actually operating computer system, not about laboratory testing of the requirements of a program not yet written, the topic addressed in the *Desgrousilliers, et al.* patent.

The Examiner next cited a passage in the *Weber* patent (col. 63, lines 54-63) which summarily described how a customer's computer system 120 entered into a secure dialog or handshake session (or linkage) 150 with a merchant's computer system 130 which in turn entered into a separate secure dialog or handshake session (or linkage) 170 with a payment gateway computer system 140 for the purpose of consummating a purchase based upon usage of a credit card, with bank approval. Again, this *Weber* passage has nothing whatsoever to do with software development and requirements testing, as taught by the Desgrousilliers et al. patent.

And finally, the Examiner cited a passage in the *Weber* patent (col. 137, line 29-63) that is taken from a section of the *Weber* patent that discusses certificate processing. The cited passage relates to encryption methods for data managed by a data manager and, in particular, discusses a "wallet" feature that "contains personal information." This, again, has nothing to do with the teachings of the *Desgrousilliers, et al.* patent relating to program requirements testing.

The second element of claim 24 clearly does not read upon any of these passages of the *Weber* patent taken either alone (as discussed above) or in combination with the above passage from the *Desgrousilliers, et al.* patent.

The Examiner further indicates that the second element of claim 24 also reads upon the three passages from the *Weber* patent (just summarized) taken in combination with the following passage taken from the *Desgrousilliers, et al.* patent:

> Knowledgebase file storage devices 13 contain a global set of testing conditions pertinent to at least the subsystem type or types to which the subsystem control facility under development is pertinent. This knowledgebase is attached hereto in microfiche form as Appendix D, which is hereby incorporated by reference. In general, the testing conditions embody a test generation strategy in which the common and minimal set of required techniques is generated. This set includes input validation and syntax

> checking, configuration, including boundary-value analysis and equivalence partitioning (i.e., identifying classes of inputs over which behavior is similar), state transitions, transactions, path, static testing by consistency verification of the requirement statements, dynamic testing of the binary table, version control, and test case--results analysis independence. In addition, the test generation strategy includes providing test script input format which is valid for use with a test SCF unit and also a coded SCF product module. The test generation strategy includes the concept of non-random test generation: viz. the same tests should be generated each time the test is run for a given subsystem; and generation of values of attributes and the sequence in which the SCF commands are generated should also be done in a non-random fashion. In addition, a cause and effect component is incorporated into the test strategy, i.e., the use of test cases investigating input condition combinations and resulting behavior. The actual testing knowledgebase includes a number of files which define all general classes, objects and properties, a utility and general rule library, and a plurality of test sets which generate specific tests, such as state/node tests (Abort, Start, Stop commands), display tests (Info, Status commands), miscellaneous tests (Boot, Load commands), attribute tests (Add, Alter, Trace commands), negative attribute tests, and other negative tests (all commands, specific, Path). The SPI tests generate negative SPI header and specific token tests, and positive combinations of specific tokens described below. [col. 5, line 59 to col. 6, line 28]

This passage is taken from a description of Figure 1 of the *Desgrousilliers, et al.* patent, a figure illustrating a "system for generating a suite of test scripts" for later use in testing out the requirements for a program "prior to preparing actual software code ...." (col. 5, lines 27-31) This particular passage explains that a global set of "testing conditions" pertinent to a network management application program (called a "subsystem control facility" or "SCF" in this passage) is available in a storage device 13 to guide a HyperCard program (Illustrated in Figures 2-14) in walking the programmer through the process of producing test scripts which, in Figure 8, is later to be used in testing out the programmer's requirements for that same program.

There is no mention in this passage of providing analyzers containing executable program steps that can compute, from a computer's configuration information, a report of any kind. The claim language clearly does not read upon this passage when considered by itself.

When this passage is viewed in combination with the *Weber* patent Abstract, which addresses the topic of how to test an operating merchant's computer system by having it communicate with a dummy bank gateway computer system, this combination of the *Weber*

patent's teachings relating to the testing of an operating computer with the *Desgrousilliers, et al.* patent's teachings relating to the testing of requirements for a program not yet written is not one that makes any sense and is also not one that has anything to do with gathering hardware and software configuration information from running computers and providing analyzers for such information, as element two of claim 24 requires. The claim language clearly does not read upon this combination of references.

When this passage is viewed in combination with column 63, lines 54-63 of the *Weber* patent, the same result follows. Lines 54-63 describe how a customer's computer system 120 enters into a secure dialog or handshake session (or linkage) 150 with a merchant's computer system 130 which in turn enters into a separate secure dialog or handshake session (or linkage) 170 with a payment gateway computer system 140 for the purpose of consummating a purchase based upon usage of a credit card, with bank approval. The *Weber* patent's teachings concerning the payment handshake sessions (or linkages) 150 and 170 have nothing whatsoever to do with preparing test scripts for software development and requirements testing, as taught by the above passage in the Desgrousilliers et al. patent. This combination of references makes no sense at all.

And finally, when this passage is viewed in combination with column 137, lines 26-63 of the *Weber* patent, a passage taken from the *Weber* patent's discussion of certificate processing, the language of the second element of claim 24 cannot be read upon this combination either. The cited passage in *Weber* discusses a "wallet" feature that "contains personal information." This, again, has nothing whatsoever to do with the software requirements testing teachings of the *Desgrousilliers, et al.* patent.

The claim language clearly does not read upon any of these passages of the *Weber* patent either taken alone (as discussed above) or when taken in combination with the above passage from the *Desgrousilliers, et al.* patent.

The Examiner finally indicates that the second element of claim 24 also reads upon the three passages from the *Weber* patent (just discussed) taken in combination with the following passage taken from the *Desgrousilliers, et al.* patent:

> With reference to the attached analyzed results sample, this sample illustrates the analysis of the INFO command using a line object. The contents of the command buffer are first set forth followed by the contents of the corresponding response buffer. Below these contents, is the analysis summary of the header command-response results analysis. As can be seen, there are no errors indicated and one warning: viz. the response MAXRESP is not the same as in the command buffer. Next follows the summary analysis of the SPI command-response results analysis, which indicates the command-response processing is valid. Next follows the analysis summary of the COM command-response results analysis, which indicates no warnings, and one error. This error is identified as missing the ZCOM-TKN-REQD token. The proper return code (RETCODE) is also indicated. There follows a second set of analyzed result for the same command which illustrates different errors and warnings, the manner in which the errors are identified and the reasoning why the identified item is an error, as well as the appropriate correction. [col. 16, lines 46-65]

This passage is a brief summary description of a report. The actual full text of this same report is presented in columns 19 to 28 of the *Desgrousilliers, et al.* patent.

Here is how this report is generated: With reference to Figure 8, it will be recalled that the *Desgrousilliers, et al.* patent teaches that prior to the writing (or coding) of a new network management application program, a programmer prepares a detailed statement of that program's requirements (subroutine calls and returns, etc.) and presents the requirements assembled into a subsystem binary table 21 (see Figure 8, and see also col. 10, lines 52-58 and col. 10, lines 65 to col. 11, line 3). The programmer also prepares test data for these requirements and presents them assembled into test scripts 22. The programmer then has a test unit 25 use the test data contained in the test scripts 22 to test or exercise the requirements as presented in the binary table 21. The test unit 25 then generates the requirements test results which are presented in a detailed test results files 29 that the programmer may then read and study directly (col. 13, lines 55 to 58). In the case of some of the more standardized test scripts representing standardized network management subroutine calls that are made by many network management applications, and not in the case of highly specialized subroutine calls unique to the new program under development, the programmer may call upon an inference engine 16 which has been preprogrammed in the artificial intelligence manner to analyze some the test results files 29 and to generate the report (col. 13, lines 60-63) a sample of which appears in columns 19 to 28 of the patent and which report is summarized by the above passage cited by the Examiner. In summary, then, the

above passage describes a report that is generated by an artificial intelligence inference engine programmed to read through program analysis requirements test results and to condense those test results down into a more readable format for presentation.

The second element of claim 24 does not read upon this passage. The second element calls for the analysis of "configuration information" (not test results) "gathered from a single computer" (there is no mention at all of collecting information from single computers in the *Desgrousilliers, et al.* patent) identifying at least one issue "relating to the computer" (not relating to some yet-to-be-written program's requirements). The claim language clearly does not read upon this passage.

When this passage is combined with the *Weber* patent's abstract, the abstract's teachings concerning connecting a merchant's computer system to a test computer, rather than the computer of an actual bank, still includes no teachings about collecting and analyzing configuration information from individual computers. The claim language does not read upon this combination of references.

When this passage is viewed in combination with column 63, lines 54-63 in the *Weber* patent, the same result follows. Lines 54-63 describe how a customer's computer system 120 enters into a secure dialog or handshake session (or linkage) 150 with a merchant's computer system 130 which in turn enters into a separate secure dialog or handshake 170 with a payment gateway computer system 140 for the purpose of consummating a purchase based upon usage of a credit card, with bank approval. Again, the *Weber* patent's teachings concerning payment handshake sessions (or linkages) 150 and 170 have nothing whatsoever to do with preparing test scripts for software requirements testing and then analyzing the requirements test results, as is taught by the above passage in the Desgrousilliers et al. patent. This combination of references makes no sense at all.

And finally, when this passage is viewed in combination with *Weber* patent passage (col. 137, line 29-63) that is taken from a section of the *Weber* patent which discusses certificate processing, the language of the second element of claim 24 cannot be read upon this combination either. The cited passage in *Weber* relates to methods for data managed by a data manager and, in particular, it discusses a "wallet" feature that "contains personal

information." This, again, has absolutely nothing whatsoever to do with the software design and testing teachings of the *Desgrousilliers, et al.* patent relative to the program's requirements.

The claim language clearly does not read upon any of these passages of the *Weber* patent either taken alone (as discussed above) or when taken in combination with the above passage from the *Desgrousilliers, et al.* patent.

g) The Third Element of Claim 24 Does Not Read Upon The Combination of *Weber* with *Desgrousilliers, et al.*

The third element of claim 24 reads as follows:

> defining at least one task definition comprising a list of one or more of said computers and a list of one or more of said analyzers;

The Examiner maintains that this third element of claim 24 reads upon three different passages in the *Weber* patent (discussed above) when combined with the following passage taken from the *Desgrousilliers, et al.* patent:

> ... [T]he user specifies by means of ... screens those types of test results for which analysis is desired. ... Once the user specifies the test results to be analyzed, the system 10 uses the knowledge bases ... and the contents of the test results files to perform the analysis.
>
> The areas of analysis are as follows. All SPI requirements are verified for a subsystem's conformance. The main ones include:
>
> Header items and data for both command and response buffers, including lengths, SSID, versions; Cross-verify command to response buffer header items and data;
>
> RETCODE, test generator positive or negative test specific;
>
> DataLists-EndLists response records, including RETCODE, object type, name, state, cross-check MAXRESP token, Z-(subsystem specific token for INFO and STATUS commands);
>
> ErrLists-EndLists response records, including RETCODE, RETCODE and error, error and SSID, object type or name;
>
> Allow and response type processing with validation;
>
> Additionally, token code and value verification is made and includes (but is not limited to) the following:
>
> Mis-match between command, object type and object name
>
> Missing token
>
> Nonexistent object or token

Not supported
Invalid format
Unsupported format (for example; wildcards)
Duplicate tokens and or values
Inconsistent (for example; modifiers, attributes)
Command-node or state behavior:
Configuration
Scenario (for example; sequence)
State transition

To start the test results analysis, the user clicks the "Result Analysis" button from the main Bartender card. The flowchart in FIG. 9 shows the position of the results analysis stack in relation to the Bartender tool stack.

When the "Result Analysis" button is clicked from the main Bartender card, the card shown in FIG. 10 appears. This screen shows the name of the subsystem where the test results, the subsystem version, and other pertinent information are analyzed.

Click the "SCF Test Results" button to move to the SCF test results card for analysis of SCF test results.

Click the "SPI Test Results" button to move to the SPI test results card for analysis of SPI test results.

Types of Test Results

The types of test results that a user can analyze are:

SCF test results
SCF test results for a specific command
SCF test results for a specific object type
SCF test results for a specific command-object type pair
User test results for a specific command-object type pair
SPI test results

For the first four SCF test results in the list, a user can specify the analysis of:

a) positive test results
b) negative test results
c) both types of test results

Once the test results to be analyzed have been specified, the tool uses the information about the subsystem to analyze the results.

[col. 13, line 64 to col. 14, line 67 – note that the first (introductory) paragraph above was included here by applicants for completeness and clarity, but this first paragraph was not actually cited by the Examiner.]

This passage teaches that an inference engine 16 (Figure 1), guided by artificial intelligence data constructs, can be called upon by a programmer to analyze some of the test results 29 generated by a test unit 25 (Figure 8) after the test unit 25 has finished testing the requirements (represented by the binary table 21) for a computer program that the programmer has not yet written, testing these requirements against test data (represented by

the test scripts 14 and 22). This passage teaches further that the computer programmer can "specify the test results to be analyzed."

In contrast to this, the third element of claim 24 calls for "task definitions" that list the analyzers which are to be called upon to perform analysis. The claim thus permits one to select which types of analysis are to be performed. Contrary to this, in the above passage, the programmer is not permitted to select which analysis is to be performed. Instead, the above passage permits the programmer to specify which "test results" are to be analyzed. The third element of claim 24 also calls for the task definitions to list the computers whose hardware and software configuration information is to be subjected to analysis. Contrary to this, the above passage calls for test results to be listed, and these test results are not configuration information and are not computed from configuration information. These test results come from testing the requirements of programs not yet written. Because this passage does not teach listing or selecting analyzers and also does not teach listing or selecting computers whose configuration information is to be analyzed by the listed or selected analyzers, the claim language clearly does not read upon this passage.

Considering this passage in combination with column 39, lines 13-19 of the *Weber* patent, the *Weber* passage explains that a merchant can alter information stored on the merchant's computer such as information specific to each credit card issuer, information specific to the merchant's acquiring bank, and also the merchant's name. The above passage taken from the *Desgrousilliers, et al.* patent teaches that a programmer testing the requirements for a yet-to-be-written program may, following testing of the requirements, select which test results data to subject to a full analysis. Clearly, these teachings of *Weber* and *Desgrousilliers, et al.* are completely unrelated and are not combinable in any meaningful sense to serve as the basis for and obvious-type rejection.

Considering this passage in combination with column 63, lines 14-24 of the *Weber* patent, a passage that discusses how a single merchant's computer system may be supplied with multiple bank-supplied terminal identifiers so that the merchant's computer can process several payment approval or capture transactions simultaneously, again these teachings of *Weber* relating to banking and those of *Desgrousilliers, et al.* presented in the above passage relating to software requirements testing are completely unrelated and are not combinable in

any meaningful sense to serve as the basis for an obvious-type rejection. And the same may be said when this passage taken from *Desgrousilliers, et al.* is considered in combination with column 64, lines 15-45 of the *Weber* patent that discusses how several merchants may share a single merchant's computer system 130, a topic that is entirely unrelated to Desgrousillier's teachings concerning software requirements testing.

The language of claim 24's third element clearly does not read upon the combination of the above passage taken from the *Desgrousilliers, et al.* patent with any of the three passages taken from the *Weber* patent that the Examiner has cited.

h) The Fourth Element of Claim 24 Does Not Read Upon *Desgrousilliers, et al.*

The Examiner indicates that the fourth element of claim 24:

> in a fully automated fashion, and guided by one or more of said task definitions --

reads upon the following passage taken from the *Desgrousilliers, et al.* patent:

> The SPI test results, on the other hand, are all automatically analyzed and the result is preferably printed out with comments. [col. 13, lines 60-62]

This passage speaks of analyzing automatically some of the "test results" generated by the process of testing the software requirements specifications for a yet-to-be-written computer program against test data, as has been explained above. This passage says this analysis is automatic, and other parts of this patent explain that an inference engine 16 (shown in Figure 1) performs this analysis, presumably using artificial intelligence strategies. (The automatically-generated report appears in columns 19 to 28.)

Entirely contrary to this, element four of claim 24 speaks of being guided in a fully automated fashion by task definitions which claim element three specifies contain lists of computers whose hardware and software configuration information is to be analyzed and which claim element three also specifies contain lists of the analyzers whose "executable program steps" are to process that configuration information – see claim element two. Clearly, the automated testing of hardware and software configuration information taken from a selected list of computers by "executable program steps" contained in a selected list of

analyzers is not the same as the automated testing of selected test results resulting from program requirements testing carried out before a computer program is written.

The Examiner further indicates that this claim language reads upon column 14, lines 5-67 of the *Desgrousilliers, et al.* patent. This passage, and the introductory paragraph preceding this passage (col. 13, line 64 to col. 14, line 5), were just discussed above, and it was shown that this passage teaches that a programmer, who has just tested out the requirements for a computer program not yet written, may take the data files resulting from that testing of requirements and automatically analyze selected ones of the test results using an inference engine.

Essentially the same rationale applies to this passage as applied to the passage just discussed above. Briefly stated, element four of claim 24 speaks of being guided in a fully automated fashion by task definitions which element three specifies contain lists of computers whose hardware and software configuration information is to be analyzed and which claim element three also specifies contain lists of the analyzers whose "executable program steps" are to process that configuration information – see claim element two. Clearly, the automated testing of hardware and software configuration information taken from a selected list of computers by "executable program steps" contained in a selected list of analyzers is not the same as the automated testing of selected test results resulting from program requirements testing carried out before a computer program is written.

The fourth element of claim 24 clearly does not read upon either of these passages taken from the *Desgrousilliers, et al.* patent.

i) The Fifth Element of Claim 24 Does Not Read Upon *Desgrousilliers, et al.*

The Examiner indicates that the fifth element of claim 24:

> harnessing each of the task definition's listed analyzers to configuration information gathered from each of the task definition's listed computers;

reads upon Figure 9 of the *Desgrousilliers, et al.* patent.

With reference to Figure 9 of the *Desgrousilliers, et al.* patent, this figure shows the outline structure of a HyperCard stack. "HyperCards" are viewable card images arranged into virtual "stacks" of cards each of which can contain one or more mouse-clickable, button-style Hypertext linkages to other cards in the same stack that enable one to jump quickly from one HyperCard card image to another and back again. Each HyperCard may also display information in scrollable windows, etc., and may also have associated with itself interpretable program code.

Figure 9 simply illustrates that a HyperCard card image (not shown in the patent) named "Bartender Main" is the topmost card in a "stack 15" (Figure 1) of such cards. Figure 9 also indicates that this topmost card, 'Bartender Main," has on its viewable card surface seven clickable buttons (not shown) that enable a computer programmer viewing this topmost card to navigate (with a click of the mouse button) to any one of seven subsidiary HyperCard images within the same stack. The seven subsidiary cards are: "Subsystem Information," "SCF PM Generator," "SCF Help and Error," "Generator Table," "SCF ES Generator," "Test Generator," and "Results Analysis.

Of these seven subsidiary card images, only two are shown and discussed to any extent in the patent. The first is the "Test Generator" HyperCard card image (shown in Figure 3), a HyperCard card image that can guide a programmer through the process of developing test scripts that can later be used to test the requirements specifications of a new, not-yet-written network management computer program which the programmer wishes to write. The second is a "Results Analysis" HyperCard card image (shown in Figure 10), a HyperCard card image that becomes useful only after requirements testing has been completed. This card guides the same programmer through the process of analyzing the files of test results 29 (Figures 1 and 8) that arise after requirements testing has been completed.

The teachings of Figure 9, cited by the Examiner, and of the related Figures 3 and 10 may be briefly summarized as follows: A programmer wishing to test out the requirements of a "network management application" computer program may use a HyperCard stack to assist the programmer in generating requirements test data by clicking down from the "Bartender Main" card to the "Test Generator" card shown in Figure 3. After requirements testing is done, the programmer may use the same HyperCard stack again to assist the

programmer in analyzing the results of testing by clicking down from the "Bartender Main" card to the "Results Analysis" card shown in Figure 10.

The Examiner maintains that the language of the fifth element of claim 24 reads upon Figure 9. With all due respect, this is simply not true. The fifth claim element calls for the analyzers listed in a task definition, including their executable program steps, to be harnessed to configuration information gathered from each of the computers also listed in the task definition. The idea here is that the executable program steps will compute from the configuration information reports of any issues that need to be drawn to management's attention. Nothing at all similar to this appears in Figure 9 of the *Desgrousilliers, et al.* patent. Figure 9 contains no reference to a task definition, particularly not to one that lists computers and that also lists analyzers. Figure 9 does not teach harnessing listed analyzers to configuration information gathered from the listed computers.

Accordingly, the fifth element of claim 9 clearly does not read upon Figure 9 of the *Desgrousilliers, et al.* patent. This, by itself, is an indication that claim 24 is clearly allowable.

j) The Sixth Element of Claim 24 Does Not Read Upon *Desgrousilliers, et al.*

The Examiner indicates that the sixth element of claim 24:

> harnessing each of the task definition's listed analyzers to configuration information gathered from each of the task definition's listed computers;

reads upon Figures 3 and 10 of the *Desgrousilliers, et al.* patent.

Figures 3 and 10 of the *Desgrousilliers, et al.* patent were just described above in overview. These two figures show the topmost card in two HyperCard stacks. Briefly summarized, "HyperCards" are virtually stackable card images which may contain one or more clickable, button-style linkages that enable one to jump from these two topmost card images to other card images and also into other programs (called as subroutines). Each HyperCard may display information, scrollable windows, etc., and each card may also contain program code.

Figures 3 illustrates a "Test Generator" HyperCard card that can guide a programmer through the process of developing test scripts that may later be used to test the programmer's requirements specification for a new, not-yet-written network management computer program. Figure 10 illustrates a "Results Analysis" HyperCard card that can guide the same programmer through the process of analyzing the files of test results 29 (Figures 1 and 8) that arise following program requirements testing.

To generate or to select test scripts, a programmer clicks one of the two buttons "SCF Tests" or "SPI Tests." Later, after requirements testing is completed, to view and evaluate the results of testing, a programmer clicks one of the two buttons "SCF Test Results" or "SPI Test Results." The difference between these two pairs of buttons will now be explained.

The term "SCF" appears in the card clickable button labeled "SCF Test" in Figure 3 and in the card clickable button labeled "SCF Test Results" in Figure 10. This term "SCF" is an abbreviation for "subsystem control facility," the name given to the general class of computer programs (also called "network management applications" – see Abstract, lines 1-2) into which general class of programs the new, yet-to-be-written program falls – the program for which requirements are to be generated and tested against test scripts. A programmer is assumed to be writing a new network management application, or "SCF." In the *Desgrousilliers, et al.* patent, The term "SCF" is also used to identify the new program's new and unique interfaces to other programs and to the user. These new interfaces are specified in the new program's requirements specification, and the programmer must be guided through the process of generating test scripts for these new and unique interfaces. The programmer initiates this process by clicking upon the "SCF Tests" button. And after requirements testing, the programmer clicks the "SCF Tests Results" button to be aided in viewing and evaluating the SCF interface requirements test results, a manual process that is not automated (col. 13, lines 59-60).

The term "SPI" appears in the card clickable button label "SPI Test" in Figure 3 and in the card clickable button label "SPI Test Results" in Figure 10. "SPI" is an abbreviation for "subsystem programmatic interface" which is a set of standardized program interfaces (or subroutine calls or function calls) that are common to and shared by many or all "network management applications." Test scripts have already been created ahead of time for any

"SPI" interface a program's requirements include – SPI test scripts do not have to be created by the individual programmer. They are simply selected by the programmer clicking upon the "SPI Tests" button. Also, after requirements testing is completed, the programmer will find that the inference engine 16 has been pre-programmed to analyze and to generate a detailed report summarizing the "SPI" compatibility testing (col. 13, lines 60-63, and see the sample report presented in columns 19-28 of the patent).

The teachings of these two figures are thus limited to teaching that a programmer wishing to test the requirements of a not-yet-written network management application computer program may use two HyperCard stacks to assist with this task. The first stack assists the programmer to generate SCF (or brand new) interface test scripts and to select SPI (or standard) interface test scripts. This first stack's top card is the HyperCard "Test Generator" which contains the clickable buttons "SCF Tests" and "SPI Tests" that the programmer may use to initiate these two tasks. The second stack assists the programmer to view the requirements test results after testing. This second stack's top card is the HyperCard "Results Analysis." This card contains the clickable button "SCF Test Results" which aids the programmer in manually browsing through the SCF (or brand new) interface requirements test results, and it also contains the button "SPI Test Results" which initiates the fully automatic generation of a report on SPI (or standardized) interface requirements test results.

The Examiner maintains that the language of the sixth element of claim 24 reads upon Figures 3 and 10. With all due respect, this is simply not true. The sixth element of claim 24 calls for the analyzers listed in the task definition and previously (see fifth element of claim 24) harnessed to hardware or software configuration information collected from computers also listed in the task definition to process the harnessed configuration information under the guidance of each analyzer's executable program steps, thereby (see claim 24 element 2) generating reports identifying issues which are then (see claim 24, element 7) utilizable for audit purposes. Nothing at all similar to this appears in Figure 3 or in Figure 10 of the *Desgrousilliers, et al.* patent. Figures 3 and 10 contain no reference to a task definition listing computers and also listing analyzers or anything in any way similar to this. Figures 3 and 10 make no mention of specific computers and clearly do not speak of harnessing listed

analyzers to configuration information gathered from listed computers. Figures 3 and 10 do not contain any reference to an analyzer's "executable program steps" and to "processing" – Figure 3 only makes mention of test scripts and data, and Figure 9 only makes mention of test result files – both forms of data, not executable program steps.

Accordingly, the language of the sixth element of claim 24 clearly does not read upon Figure 3 or upon Figure 10 of the *Desgrousilliers, et al.* patent, and on this basis alone claim 24 is clearly allowable. The same may be said of the corresponding lines 3-5 of the sixth element of claim 31 and of the corresponding lines 4-7 of the seventh element of claim 38.

For all of the above reasons, the combination of the *Weber* and *Desgrousilliers, et al.* patents do not render obvious claim 24 of the present application. Accordingly, allowance of claim 24, and of the remaining Group 1 claims 25, 31-32, 38, and 40 which are similar in scope to claim 24, is respectfully requested.

2. Group 2 Claims – Claim 27 is Patentable Over The Combination of *Weber* and *Desgrousilliers, et al.* (Claims 26, 33-34, and 39 Are Also Patentable for the Same Reasons)

Method dependent claim 27 reads as follows:

> 27. A method in accordance with claim 24 wherein the processing step further includes generating, along with at least some issue identifying reports that are generated, the identity of the computers whose configuration information was processed to generate those reports such that both the issue identifying reports and the identity of the computers may be utilized.

This claim says that the issue identifying reports are supplemented with information identifying the computer upon which each issue arose, and hence the identity of the computer that gave rise to an issue can later be inserted into reports to management along with a report of the specific issue. This computer identity information is information which the analyzers never receive and do not process – the analyzers are very simple programs which are harnessed to configuration data and which identify issues without having to be concerned with the identity of the computer from which the configuration information was gathered. This claim requires the harness mechanism, which is aware of the identity of the computer, in every case where an issue identifying report is produced, to retain information identifying the computer along with the issue identifying report which arose from processing configuration

information collected from that same computer. See Figure 21 of the present application, where the issue XML output 2124 of the analyzers 2110, 2112, and 2114 is supplemented with the identity of the computer before it is placed into the issue XML report 2126. This is done by the analyzer loader 2102. See paragraph [0187] (page 43, line 27 to page 44, line 26) of the present application, where this is fully explained in detail. This is a significant feature of the present invention which simplifies greatly the task of writing the executable program steps of an analyzer, keeping the analyzer programs very short and simple, since the analyzers do not have to process any computer identity information.

The Examiner has cited against this claim the *Desgrousilliers et al.* patent, and in particular, Figure 1, and more specifically items 18 (a mechanism for transferring files "FTP"), 29 (a file containing the results of requirements testing), 15 (a stack of HyperCards used to guide the definition of test data for program requirements testing, as shown in Figure 3, and to guide the analysis of program requirements test results, as shown in Figure 10), and 31 (a knowledge base that somehow guides the process of reviewing the results of program requirements testing).

None of this is relevant to the present invention. With reference to both Figures 1 and 8 of the *Desgrousilliers, et al.* patent, a program's requirements, represented by a subsystem binary table 21, is tested by a test unit 25 against test scripts, and the results of this testing are placed in the test results file 29 where they may be reviewed with the aid of the HyperCard stacks 15. This has nothing whatsoever to do with performing tests upon configuration information gathered from computers that are operating normally in the field and reporting issues to management. In particular, this does not teach capturing the name of a computer whose configuration information is evaluated by an analyzer, and making that computer name information available for use in reports along with any reports of issues that the analyzer may produce, which is the focus of claim 27. Figure 1 of the *Desgrousilliers et al.* patent is simply not relevant to any of this claim language.

The Examiner also cites Figures 3 and 10 of the *Desgrousilliers et al.* patent. These two figures were fully discussed above in the context of the sixth element of claim 24 and were shown to have no relevance to the present invention. In very brief summary, Figures 3 and 10 show the topmost card in two HyperCard stacks. Figure 3 illustrates a "Test

Generator" HyperCard card that can guide a programmer through the process of developing test scripts that may later be used to test the programmer's requirements specification for anew, not-yet-written network management computer program. Figure 10 illustrates a "Results Analysis" HyperCard stack that can guide the same programmer through the process of analyzing the files of test results 29 (Figures 1 and 8) that arise following program requirements testing. Neither figure teaches having a harness mechanism preserve the name of the computer whose configuration information is processed by an analyzer so that if the analyzer generates any reports of issues that have arisen, the computer's name may be added to such reports. These two figures thus do not teach the additional requirements that are set forth in dependent claim 27.

More generally, the *Desgrousilliers, et al.* patent has no teaching whatsoever that an analyzer's executable program steps should identify an issue by processing hardware or software configuration information taken from a computer, and that the results of that computation should be stored along with the identity of the computer that gave rise to the issue. Computer identity is never discussed in the *Desgrousilliers, et al.* patent, which focuses only upon analyzing the requirement's specification for a single computer program and not upon analyzing configuration information gathered from plural computers.

The claim language of claim 27 clearly does not read upon Figures 1, 3, and 10 of the *Desgrousilliers, et al.* patent.

3. Group 3 Claims – Claim 28 is Patentable Over The Combination of *Weber* and *Desgrousilliers, et al.* Claims 29-30, 35-37, and 41-42 Are Also Patentable for the Same Reasons

Claim 28 adds another major feature of the invention to the claims. As is shown in Figure 8 of the present application, the issue identifying reports (introduced in the Group 1 claims), together with the identity of each computer whose configuration information was processed to generate those reports (introduced in the Group 2 claims) are now (claim 28) used as data to guide in the processing of audit report templates 204 into one or more audit reports. Claim 28 also requires that the task definition, which already lists the analyzers and the computers whose configuration information the analyzers are to process, further include a list of the report templates, as is shown at 814 in Figure 8. Thus, extremely simple analyzer

executable program steps and a harness mechanism which applies the analyzers to configuration information and keeps track of from which computers that information came are now supplemented with more sophisticated report templates that are able to guide the production of intelligible reports identifying in human understandable terms which issues arose on what machines.

The Examiner has broken the language of claim 28 into three parts, which are discussed separately here.

a) The First Element of Claim 28 Does Not Read Upon *Desgrousilliers, et al.*

The Examiner first maintains that the first element of claim 28, which introduces "report templates" as an element of the invention:

providing a plurality of audit report templates;

reads upon col. 4, lines 12-14 of the *Desgrousilliers, et al.* patent. The Examiner cited only a single sentence, and it is ***italicized*** in the longer passage that is reproduced just below. (Applicants have reproduced portions of a preceding sentence to clarify the meaning of the word "uses" which appears in the ***italicized*** sentence actually cited by the Examiner):

> From a process standpoint, the invention comprises [col. 3, lines 52-53] ... using the first and second knowledgebases to generate a suite of test scripts for testing the proposed set of requirements prior to coding. [col. 3, line 66 to col. 4, line 1] ... ***The step of using includes the step of generating test scripts and corresponding templates for mapping the test scripts into a subsystem.*** [col. 4, lines 12-14 of the *Desgrousilliers, et al.* patent]

This passage in the *Desgrousilliers, et al.* patent is immediately followed by another passage which the Examiner has cited against similar claim language relating to "audit report templates" and appearing in the claims 29 and 30. It is thus appropriate to reproduce that passage at this point in the discussion. The final sentence of this lengthy passage, the only sentence mentioning "templates," is also ***italicized*** to make it stand out:

> From an apparatus standpoint, the invention comprises a system for preparing a suite of test scripts for testing a proposed subsystem control facility set of requirements prior to coding the proposed design, the system

> including first knowledgebase containing the rules governing the operation of a managed database network and a library of permitted commands and objects, second knowledgebase containing test generation information relating to the commands and objects specific to the proposed subsystem control facility set of requirements, a user interface coupled to the knowledgebase for permitting selection of types of tests and specific commands and objects to be tested, and test script generator coupled to the first and second knowledgebases for generating a suite of test scripts from the first and second knowledgebases for use in testing the proposed set of requirements prior to coding. The library contained in the first knowledgebase includes a global set of object types, object, permitted object attributes and permitted object values. The second knowledgebase containing test generation information includes a set of common and minimal required test techniques, including positive tests for testing the ability of the proposed subsystem control facility to process valid commands, objects and other data, and negative tests for testing the ability of the proposed subsystem control facility to process invalid commands, objects and other data. ***The test script generator includes a portion for generating a plurality of test script files and corresponding template files.*** [col. 4, lines 15-41]

To determine the meaning of "template" which appears in the two ***italicize*** passages set forth above, one must review and study other portions of the *Desgrousilliers, et al:* patent.

A later passage (col. 9, line 44 to col. 10, line 41) explains that these templates are needed to present the test script data in different ways in different testing contexts (col. 10, lines 40-41).

> ... The purpose of the template file is to provide ... :
> Portable test scripts to systems with different addresses, process names, configuration requirements.
> Adjustable test scripts for subsystem or platform sensitive data. [col. 9, lines 46-51]

But this special context of using templates to prepare test data for presentation in machine understandable form for submission to various automated test regimens on differing platforms is far different than the context of using templates to transform issue identification brief reports and the names of the computers where those issues arose into meaningful, human understandable audit reports. Thus, this passage taken from the claim clearly does not read upon either of the two passages quoted above which were cited by the Examiner.

The Examiner cited the following passage taken from the *Desgrousilliers, et al.* patent against elements of claim 40 that correspond generally to this first element of claim 28 in that they relate to report templates:

> In the preferred embodiment, a Macintosh computer with at least eight megabytes of memory and system 6.05 or newer is used as the platform for system 10. HyperCard stacks 15 (HyperCard is a trademark of Apple Computer Company) comprise a graphical interface product available from Apple Computer Company which runs on the Macintosh computer platform. HyperCard version 2.0 or newer is preferred, and the HyperCard application memory size is preferably at least 5500 megabytes. These stacks are used as a "front end" to developmental system 10 and provide the primary interface for the user. Artificial intelligence inference engine 16 is used as the "back end" to developmental system 10 and preferably comprises Nexpert Object, an artificial intelligence inference engine available from Neuron Data. Inference engine 16 is used to generate the test scripts, and to analyze the testing results once the testing has been completed. [col. 6, lines 32-48]

This passage does not mention templates at all, and accordingly is not relevant to the claim language. The word "template" does appear elsewhere in the *Desgrousilliers, et al.* patent, but it is used only in the context of preparing test data, and not in the context of generating any kind of report (for example, see col. 4, lines 39-41 of the *Desgrousilliers, et al.* patent, which are set forth above in this subsection).

Accordingly, the first element of claim 28 does not read upon these passages of the *Desgrousilliers, et al.* patent

b) The Second Element of Claim 28 Does Not Read Upon *Desgrousilliers, et al.*

The Examiner next addresses the second element of claim 28:

> defining the at least one task definition to further comprise a list of one or more audit report templates; [claim 28, second element]

This says that the task definition, in addition to listing the computers whose configuration information is to be analyzed and the analyzers that are to perform this analysis, also lists the audit report templates which are to be used for report generation.

The Examiner maintains that this claim language reads upon another passage taken from the *Desgrousilliers, et al.* patent. This passage reads as follows:

> ... The second knowledgebase containing test generation information includes a set of common and minimal required test techniques, including positive tests for testing the ability of the proposed subsystem control facility to process valid commands, objects and other data, and negative tests for testing the ability of the proposed subsystem control facility to process invalid commands, objects and other data. The test script generator includes a portion for generating a plurality of test script files and corresponding template files. [col. 4, lines 33-41]

In this passage, the "proposed subsystem control facility" is the not-yet-written program whose requirements are to be tested in an automated fashion against test data. This passage describes a "second knowledgebase" (13 in Figure 1) and explains in summary fashion that the novel "test generator" being described is guided by this knowledgebase in the generation of "test script files and corresponding template files." These template files are used as was just explained above to prepare the test data for use in possibly different contexts. Accordingly, a claim which speaks of templates for transforming computer names and issue identification reports into human-understandable audit reports does not read upon this use of templates to massage test data for presentation in different testing environments.

The second element of claim 28 clearly does not read upon this passage.

c) The Third Element of Claim 28 Does Not Read Upon the *Weber* Patent

Finally, the Examiner addresses the third element of claim 28:

> following the storing step, and guided by the task definition's listed audit report templates and by any issue identifying report generated during the processing step, generating one or more audit reports.

This says that "audit reports" (of the hardware and software configurations of computers --- see the preamble of independent claim 24) are to be generated "guided by the task definition's listed audit report templates and by any issue identifying report generated" during the harnessing of analyzer executable code to the configuration data under test.

The Examiner maintains that this claim language reads upon two passages taken from the *Weber* patent. The first passage taken from the *Weber* patent is:

> The merchant could utilize any other computer attached to the Internet utilizing a SSL or SET protocol to query the remote vPOS system and obtain capture information, payment administration information, inventory control information, audit information and process customer satisfaction information. (col. 64, lines 1-7)

This passage, as was explained above, relates to financial auditing of bank payment transaction records, as is clear from the context of this passage, and not audit reports used "for auditing the software or hardware configurations of a plurality of computers in one or more enterprises" as the claim 24 preamble and other claim elements require. The first element of the claim requires the collection of configuration information from computers relating to their hardware and software configurations, for example. The claim language clearly does not read upon this passage which relates to bank transaction auditing records.

The second passage taken from the *Weber* patent is:

> Gateway statistics about transaction requests (by transaction type) and transaction results (e.g., success, failed due to host, failed due to authentication, etc.) can be determined at any time for a particular time interval by generating a report. [col. 124, lines 14-17]

The "transaction requests" discussed here are requests by a merchant's computer made to the merchant's bank's computer to request authorization to extend credit to a particular customer and, later on, to request capture of that payment. These transactions are payment transactions. They have nothing whatsoever to do with the hardware and software configuration of plural computers. There is also no mention here of the use of templates for report generation purposes in this passage and elsewhere in the *Weber* patent. Once again, the claim language clearly does not read upon this passage of the *Weber* patent.

Accordingly, the third element of claim 28 does not read upon either of these passages taken from the *Weber* patent.

I. Conclusion – The Claims 24 to 42 Are Not Rendered Obvious by The Combination of the *Weber* and *Desgrousilliers, et al.* Patents

The above discussion has demonstrated that all the claims are patentable over the combination of these two patents. All the passage and figures cited by the Examiner have been discussed, and none of them disclose the requirements set forth in the claim elements against which they have been cited. For all the reasons stated above, all the claims are now believed to be in condition for allowance. Accordingly, reversal of the Examiner's decision and allowance of the claims 24 to 42 is respectfully requested.

I. Deposit Account Charge Authorization

The Commissioner is hereby authorized to charge any additional fees which may be required regarding this application under 37 C.F.R. §§ 1.16-1.17, or credit any overpayment, to Deposit Account No. 06-1450. Should no proper payment be enclosed herewith, as by a check being in the wrong amount, unsigned, post-dated, otherwise improper or informal or even entirely missing, the Commissioner is authorized to charge the unpaid amount to Deposit Account No. 06-1450. If any extensions of time are needed for timely acceptance of papers submitted herewith, Applicants hereby petition for such extension under 37 C.F.R. §1.136 and authorizes payment of any such extensions fees to Deposit Account No. 06-1450.

Respectfully submitted,

Date Dec. 6, 2004

By _____

FOLEY & LARDNER LLP
321 North Clark Street
Chicago, Illinois 60610

Telephone: (312) 832-4596
Facsimile: (312) 832-4700

James A. Sprowl
Attorney for Applicants
Registration No. 25,061
Telephone: (847) 446-7399

VIII. CLAIM APPENDIX

Listing of Claims and their Status:

1 to 23. (Cancelled)

24. (Rejected) A computerized method for auditing the software or hardware configurations of a plurality of computers in one or more enterprises, comprising the steps of:

in a fully automated manner, collecting from each of a plurality of networked computers configuration information defining each computer's software configuration or hardware configuration or both;

providing a plurality of analyzers based on expert knowledge, each analyzer comprising the executable program steps needed to compute, from selected configuration information gathered from a single computer, a report identifying at least one issue relating to the computer;

defining at least one task definition comprising a list of one or more of said computers and a list of one or more of said analyzers; and

in a fully automated fashion, and guided by one or more of said task definitions --

harnessing each of the task definition's listed analyzers to configuration information gathered from each of the task definition's listed computers;

processing the configuration information so harnessed under the guidance of each analyzer's executable program steps; and

utilizing any issue identifying report generated during the processing step for audit purposes.

25. (Rejected) A method in accordance with claim 24

wherein the collecting step includes placing this configuration information into a tracker database from which configuration information is later retrieved during the harnessing or processing steps;

wherein the providing analyzers step includes placing these analyzers into an analyzer database from which analyzers are later retrieved during the harnessing or processing steps; and

wherein the processing step includes storing any issue identifying report generated in an issues database from which it may later be retrieved and utilized.

26. (Rejected) A method in accordance with claim 25 wherein the processing step further includes storing at least some issue identifying reports generated in an issues database together with the identity of the computers whose configuration information was processed to generate those reports and from which issues database both the issue identifying reports and the identity of the computers may later be retrieved and utilized.

27. (Rejected) A method in accordance with claim 24 wherein the processing step further includes generating, along with at least some issue identifying reports that are generated, the identity of the computers whose configuration information was processed to generate those reports such that both the issue identifying reports and the identity of the computers may be utilized.

28. (Rejected) A method in accordance with claim 27 which further includes:

providing a plurality of audit report templates;

defining the at least one task definition to further comprise a list of one or more audit report templates; and

following the storing step, and guided by the task definition's listed audit report templates and by any issue identifying report generated during the processing step, generating one or more audit reports.

29. (Rejected) A method in accordance with claim 28

wherein the collecting step includes placing this configuration information into a tracker database from which configuration information may later be retrieved during the harnessing or processing steps;

wherein the providing analyzers step includes placing these analyzers into an analyzer database from which analyzers are later retrieved during the harnessing or processing steps;

wherein the providing audit report templates step includes placing these templates into a report template database from which they may later be retrieved during the audit report generating step; and

wherein the utilizing step includes storing any issue identifying report generated in an issues database from which it may later be retrieved and utilized during the audit report generating step.

30. (Rejected) A computerized method for auditing the software or hardware configurations of a plurality of computers in one or more enterprises, comprising the steps of:

in a fully automated manner, collecting from each of a plurality of networked computers configuration information defining each computer's software configuration or hardware configuration or both and placing this configuration information into a tracker database;

providing a plurality of analyzers based on expert knowledge, each analyzer comprising the executable program steps needed to compute, from selected configuration information gathered from a single computer, a report identifying at least one issue relating to the computer, and placing these analyzers into an analyzer database;

providing a plurality of audit report templates, and placing these templates into a report template database;

defining at least one task definition comprising a list of one or more of said computers, a list of one or more of said analyzers, and a list of one or more of said audit report templates; and

in a fully automated fashion, and guided by one or more of said task definitions --

harnessing each of the task definition's listed analyzers to configuration information gathered from each of the task definition's listed computers;

processing the configuration information so harnessed under the guidance of each analyzer's executable program steps;

storing any issue identifying reports generated during the processing step in an issues database together with, in at least some instances, the identity of the computers whose configuration information was processed to generate these reports; and

guided by the task definition's listed audit report templates and by any issue identifying reports and computer identities stored in the issues database, generating at least one or more audit reports.

31. (Rejected) A computerized system for auditing the software or hardware configurations of a plurality of computers in one or more enterprises, the system comprising:

a plurality of computers each computer having software and hardware components configured in a variety of measurable ways;

a network interconnecting said computers;

a plurality of collectors, installed on at least one computer, gathering from said computers configuration information defining each computer's hardware configuration or software configuration or both;

a plurality of analyzers, each analyzer comprising the executable program steps needed to compute, from selected configuration information gathered from a single computer, a report identifying at least one issue relating to the computer;

at least one machine-readable task definition comprising a list of one or more computers and a list of one or more analyzers; and

an analyzer harness guided by said task definition through the process of sequentially harnessing each of the task definition's listed analyzers to configuration information gathered from each of the task definition's listed computers and executing each analyzer's executable program steps upon the harnessed configuration information, thereby generating at least some issue identifying reports.

32. (Rejected) A system in accordance with claim 31 further comprising:

a tracker database into which said collectors place said gathered configuration information and from which said analyzer harness obtains said configuration information;

an analyzer database in which said analyzers reside and from which said analyzer harness obtains said analyzers; and

an issues database into which said analyzer harness places any generated issue identifying report.

33. (Rejected) A system in accordance with claim 32

wherein said analyzer harness, in addition to generating issue identifying reports, further generates along with at least some of said reports information identifying the computers whose configuration information was harnessed to generate those issue identifying reports; and

wherein said analyzer harness places both said any generated issue identifying reports and said information identifying computers into said issues database.

34. (Rejected) A system in accordance with claim 31 wherein said analyzer harness, in addition to generating issue identifying reports, further generates along with at

least some of said issue identifying reports information identifying the computers whose configuration information was harnessed to generate those issue identifying reports.

35. (Rejected) A system in accordance with claim 34 further comprising:

audit report templates;

a list of one or more audit report templates comprising an additional part of said machine-readable task definition; and

a report generator guided by said task definition through the process of transforming at least one of the task definition's listed audit report templates into one or more audit reports, guided by any issue identifying reports and any information identifying computers generated by the analyzer harness.

36. (Rejected) A system in accordance with claim 35 further comprising:

a tracker database into which said collectors place said gathered configuration information and from which said analyzer harness obtains said configuration information;

an analyzer database in which said analyzers reside and from which said analyzer harness obtains said analyzers;

a report template database containing said audit report templates and from which said report generator obtains said audit report templates; and

an issues database into which said analyzer harness places any generated issue identifying report and any information identifying computers, and from which issues database said report generator obtains issue identifying reports and information identifying computers.

37. (Rejected) A computerized system for auditing the software or hardware configurations of a plurality of computers in one or more enterprises, the system comprising:

a plurality of computers each computer having software and hardware components configured in a variety of measurable ways;

a network interconnecting said computers with a tracker database;

a plurality of collectors, installed on at least one computer, gathering from said computers configuration information defining each computer's hardware configuration or software configuration or both, and storing the gathered configuration information in the tracker database;

a plurality of analyzers residing within an analyzer database, each analyzer comprising the executable program steps needed to compute, from selected configuration information gathered from a single computer, a report identifying at least one issue relating to the computer;

audit report templates residing within a report templates database;

at least one machine-readable task definition comprising a list of one or more computers, a list of one or more analyzers, and a list of one or more audit report templates;

an analyzer harness connecting to said tracker database and to said analyzer database and guided by said task definition through the process of sequentially harnessing each of the task definition's listed analyzers to configuration information gathered from each of the task definition's listed computers, executing each analyzer's executable program steps upon the harnessed configuration information, and storing any issue identifying reports generated during this execution in an issues database together with, in at least some instances, the identity of the computers whose configuration information was harnessed to generate these issue identifying reports; and

a report generator connecting to said issues database and to said report template database and guided by said task definition through the process of transforming at least one of the task definition's listed audit report templates into one or more audit reports, guided by any issue identifying reports and computer identities stored in the issues database.

38. (Rejected) A computerized system for auditing the software or hardware configurations of a plurality of computers in one or more enterprises, the system comprising:

a plurality of computers each computer having software and hardware components configured in a variety of measurable ways;

tracker database means, analyzer database means, and issues database means all for storing digital information;

a network interconnecting said computers with said tracker database means;

collector means, installed on at least one computer, for gathering from said computers configuration information defining each computer's hardware configuration or software configuration information or both, and for storing this gathered configuration information in said tracker database means;

analyzer means, residing within said analyzer database means, for defining the executable program steps needed to compute, from selected configuration information gathered from a single computer, a report identifying at least one issue relating to the computer;

machine-readable task definition means for defining a list of one or more computers and a list of one or more analyzer means; and

analyzer harness means connecting to said tracker database means, said analyzer database means, and said issues database means for sequentially harnessing, under the guidance of said task definition means, each of the task definition mean's listed analyzer means to configuration information gathered from each of the task definition mean's listed computers, for executing each analyzer mean's executable program steps upon the harnessed configuration information, and for storing any issue identifying report generated during this execution in said issues database.

39. (Rejected) A system in accordance with claim 38

wherein the analyzer harness means, in addition to storing any issue identifying report generated during program step execution in said issues database means, also stores in said issues database means along with at least some issue identifying reports the

identities of the computers whose configuration information was harnessed to generate these issue identifying reports.

40. (Rejected) A system in accordance with claim 38 further comprising:

report template database means for storing digital information;

audit report template means, residing within said report template database means, for defining all or portions of one or more audit reports;

wherein the task definition means further comprises means for defining a list of audit report template means; and

report generator means, connecting to said issues database means and said report template database means, for generating at least one audit report, guided by the task definition mean's listed report templates and by the issue identifying reports stored in the issues database.

41. (Rejected) A system in accordance with claim 40:

wherein the analyzer harness means, in addition to storing any issue identifying report generated during the program execution step in said issues database means, also stores along with at least some of the issue identifying reports the identities of the computers whose configuration information was harnessed to generate those issue identifying reports; and

wherein the report generator means is also guided by the computer identities stored in the issues database.

42. (Rejected) A computerized system for auditing the software or hardware configurations of a plurality of computers in one or more enterprises, the system comprising:

a plurality of computers each computer having software and hardware components configured in a variety of measurable ways;

tracker database means, analyzer database means, report template database means, and issues database means all for storing digital information;

a network interconnecting said computers with said tracker database means;

collector means, installed on at least one computer, for gathering from said computers configuration information defining each computer's hardware configuration or software configuration or both, and for storing this gathered configuration information in said tracker database means;

analyzer means, residing within said analyzer database means, for defining the executable program steps needed to compute, from selected configuration information gathered from a single computer, a report identifying at least one issue relating to the computer;

audit report template means, residing within said report template database means, for defining all or portions of one or more audit reports;

machine-readable task definition means for defining a list of one or more computers, a list of one or more analyzer means, and a list of one or more audit report template means;

analyzer harness means connecting to said tracker database means, said analyzer database means, and said issues database means for sequentially harnessing, under the guidance of said task definition means, each of the task definition mean's listed analyzer means to configuration information gathered from each of the task definition mean's listed computers, for executing each analyzer mean's executable program steps upon the harnessed configuration information, and for storing any issue identifying reports generated during this execution to in said issues database together with, in at least some instances, the identity of the computers whose configuration information was harnessed to generate these issue identifying reports; and

report generator means, connecting to said issues database means and said report template database means, for generating at least one audit report, guided by the task

definition mean's listed report templates and by the issue identifying reports and computer identities stored in the issues database.